CASIO

G005-07383C

触って・覚えて 実務に役立つ

実践パソコン必勝法

監修●渡辺　茂
著者●柿園昭俊
編集製作●技術評論社

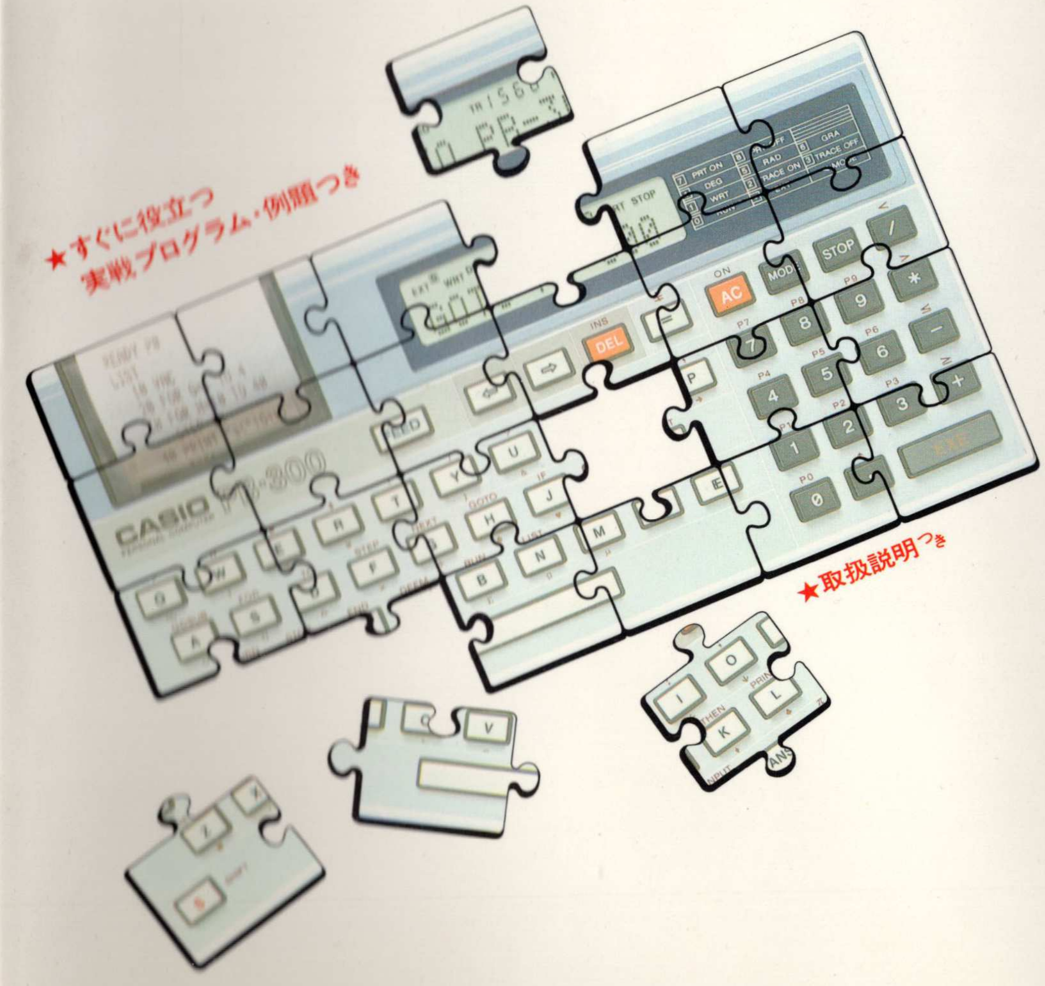
CASIO

実践パソコン必勝法

触って・覚えて・実務に役立つ　カシオパソコンPB-300

監修／渡辺　茂　著者／柿園昭俊　編集製作／技術評論社

★すぐに役立つ実戦プログラム・例題つき



★取扱説明つき

カシオ計算機株式会社

CASIO

触って・覚えて
実務に役立つ
実践パソコン必勝法
監修●渡辺　茂
著者●柿園昭俊
編集製作●技術評論社
CASIO

実践パソコン必勝法

触って・覚えて・実務に役立つ　カシオパソコンPB-300

監修／渡辺　茂　著者／柿園昭俊　編集製作／技術評論社

カシオ計算機株式会社

実践パソコン必勝法

触って・覚えて・実務に役立つ　カシオパソコンPB-300

監修／渡辺 茂　著者／柿園昭俊　編集製作／技術評論社

カシオ計算機株式会社

監修のことば

パーソナルコンピュータに対するまなざしは，ますます熱を帯びてきている昨今です．

ユーザーサイドでも，ゲームを楽しむ若い人達と，一方ではビジネス，研究開発やグラフィックなど，実務に利用したいと考える方々も多くなってきました．

各メーカーでも，多様なニーズに対応して次々と新しい製品を発表しています．16ビットのものからハンドヘルドタイプ，ポケッタブルなものまで，さまざまです．

ところで，カシオ計算機㈱では，昨年9月1万円台のパソコンPB－100を発売し，それとともに「パソコン必勝法」を作りました．これはBASICの入門書でしたが，本書はこれに続くもので，BASICをある程度会得された方々に対し，さらに自分でBASICのプログラミングをやり，楽しく，有効にパソコンを使えるようにと願って書かれたものです．

内容は3つに分かれ，第1編はPB－300の使い方について，また第2編では，プログラム作成の手順と，各コマンドの説明，よく使われるサブルーチンについて，そして最後に生活，レジャー，ビジネス，統計など各分野別に，実際的なプログラム例をあげました．これらは皆さんの身の回りに応用されても，十分お役に立つものと思います．そして，巻末には，コマンド一覧表，キャラクター表，索引などをつけ，読む方にとって便利なものになっています．

PB－300は，PB－100のメモリーを拡張し，プリンタをつけたもので，より実用的なものになっています．

是非皆さんが，この本によってより深くBASICを理解され，自在にプログラムを組み，各分野で大いにパソコンを利用されることを望んでおります．

58年2月　渡辺 茂

読者の皆さんに

「こんな形のパソコンもあるのか，これなら使える！」

PB−300を初めとする小型のパソコンに対する素直な感想です．

「これなら使える！」という驚きの声には，二つの意味が含まれています．

第一に簡単に覚えられること，第二に実用的でもあるということです．

かつてのように，コンピュータで処理できる業務が限られており，データはコンピュータの設置場所まで届けられるという時代であれば，一部屋をすっかり専有するコンピュータで良かったかもしれません．

しかし現在は，業務の専門化，分業化，OA化が進み，さまざまな情報やデータがいろいろな形で発生しています．そこでコンピュータの分業化も必然的に起こります．

たとえば，企業の中枢となる処理や多くの企業での共同利用には汎用コンピュータを使い，一事業所，支店，店舗ごとの処理にはオフコンを用い，一部課，単位業務ごとの処理にはデスクトップ型のパソコンを当てるという具合に，それぞれの業務規模に合わせて振り分けます．

そんな中でポッカリ穴の空く分野に必要とされるのが，デスクトップ型パソコンの前段階として手軽に操作できる練習機であり，各人の仕事，単発で発生する業務，外出先でのデータ処理などにすぐ対応できる実用機です．

このような分野にふさわしいコンピュータがPB−300等の「これなら使える」小さなパソコンなのです．

命令語の数は少ないので，マニュアル片手でなくても，プログラムを思いついたその場で簡単に組むことができます．データは発生した場所で入力できます．持ち歩いてもじゃまにならない大きさです．

本書では，このような特質を生かし，限られたメモリー容量，命令をフルに機能させることができるよう，標準的な使い方から凝った使い方まで，いろいろな試みをまとめました．

いくらかでも皆様のお役に立つことができるなら幸いです．

最後に，本書の執筆に当たり，何人かの方にプログラム作成のお力添えをいただき，またカシオ計算機，技術評論社の多くの方々に御協力をいただきました．ここに厚くお礼申し上げます．

1983年2月　著者

第1編　PB-300　操作編

第2編　BASICプログラミング編

12．便利なサブルーチン集

第3編 実用ライブラリー編

第4編 巻末参考編

第1編

PB-300操作編

1 ご使用の前に

この計算機は，カシオの高度な電子技術と品質管理のもとで，厳重な検査工程を経て，皆様のお手もとに届けられています．本機を末ながくご愛用いただくために，次の点にご留意のうえ，お取り扱いください．

1 ご使用上の注意

- 計算機は精密な電子部品で構成されていますので，絶対に分解しないでください．また，投げたり落したり等のショックや，急激な温度変化を与えないでください．特に高温の所，湿気やホコリの多い所に放置したり保管することはしないでください．
  なお，温度が低いときは表示の応答速度が遅くなったり，点灯しなくなることがありますが，通常の温度になると正常にもどります．
- オプション用コネクターには，本機専用のオプション（FA−3）以外は接続しないでください．
- プリンタ用の充電池に充電中は，計算機の電源をOFFにしてください．
- 演算中は"－"を表示し，この間のキー操作は一部のキー（AC，STOP）を除いて無効ですから，常に表示を確認しながら，確実にキーを押してください．
- 電池は，計算機を使わない場合でも2年に1度は交換してください．特に消耗ずみの電池を放置しておきますと，液もれをおこし，故障等の原因になりますので，計算機内には絶対に残しておかないでください．
- 計算機のお手入れは，シンナー・ベンジン等の揮発性液体をさけ，「乾いた柔かい布」あるいは，「中性洗剤液に浸し固くしぼった布」でおふきください．

2 保証・アフターサービス

- 保証は，同封の保証書の内容によりますので，よくお読みのうえ，記入事項を確認して，大切に保管してください．
- 万一故障したときは　**❶お買い上げ店**　**❷カシオ計算機サービスセンター**のうち，ご都合のよい所へ必ず保証書をそえて，ご持参またはご郵送ください．この場合，故障内容を具体的にお知らせください．
- 修理依頼される前には，この説明書をもう一度お読みになると共に，電源の状態および，プログラムミス，操作ミスがないかをよくお調べください．
- ご不明の点やご質問，お問い合わせは，最終ページ記載のカシオ計算機へご連絡ください．

3 電源および電池交換について

本機は，計算機用としてリチウム電池《CR－2032》2個を，プリンタ用としてニッカド充電池を電源とします．

コントラストのボリウム（25ページ参照）を濃くなる位置にしても表示が薄く見えるときは電池が消耗していますので，早目に下記に示すように電池交換をしてください．

また，プリントスピードが遅くなったり，印字が薄くなったときは早目に充電を行なってください．

なお，計算機が正常に使用できても，2年に1度は必ずリチウム電池を交換してください．

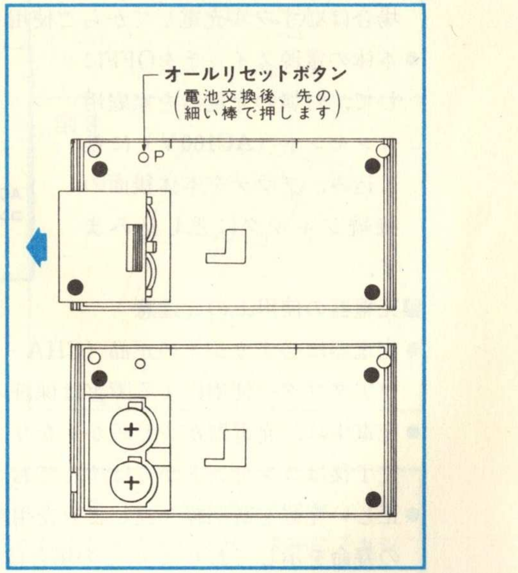
■電池交換の仕方

①電源スイッチを切ってから，電池ブタをスライドさせてはずします．

②古い電池を2個とも取り出します．（電池ボックスを下に向けて軽くたたけば簡単にはずれます．）

③新しい電池の表面を乾いた布でよくふいてから⊕側を上にして入れます．

④電池ブタをスライドさせてとじ，電源スイッチをONにしてから**リセットボタンを先の細い棒で押してください．**

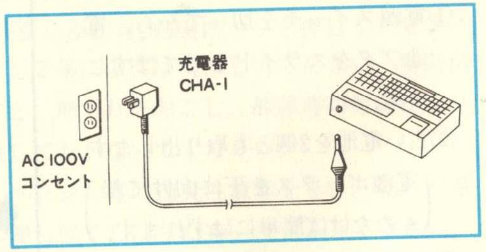


※リチウム電池は必ず2個とも交換してください．
※消耗ずみの電池は絶対に火中に投入しないでください．破裂することがあり，非常に危険です．
※電池の⊕⊖は絶対にまちがえないようにしてください．
電池は，幼児の手のとどかないところに保管してください．
万一，飲み込んだ場合にはただちに医師と相談してください．

■充電の仕方

プリンタ部は充電式電池を使用しています．プリントスピードや紙送りが遅くなったり，印字が薄くなったり印字されないときは，充電式電池の電圧が低下していますので，速やかに充電を行なってください．

充電器を接続して約15時間前後でフル充電出来ます．充電は除々に行なわれますので，充電器を接続して1～2時間後にご使用になりたいときにもご使用頂けますが，充電途中で充電器をはずしてご使用になると，印字可能行数はその分だけ少なくなりますので(フル充電のときで約3,000行)，お急ぎでない場合は必ずフル充電してからご使用下さい．



● 本体の電源スイッチをOFFにして，付属の充電器を家庭用コンセント（AC100V）に差し込み，プラグを本体後面の接続ジャックに差し込みます．

■充電器の使用上のご注意

● 充電器は必ずカシオ純正品（CHA－1）をご使用ください．他の充電器およびアダプター使用による障害は保証いたしかねます．
● 充電中は，充電器が少し暖かくなりますがご心配ありません．しかし，充電完了後はコンセントからはずしておいてください．
● 正しい充電を数回繰り返しても使用時間が半減してきたときは，充電式電池の寿命を示しております．この場合は最終ページ記載のサービスセンターにお申しつけいただければ，有償にてお取り替えいたします．

2　各部の名称とそのはたらき

4 ロールペーパーのつけ方

①電源スイッチをONにします．
②プリンタカバーを図1のようにあけます．
③ロールペーパーの先端をまっすぐに切って，ロールペーパーの先端が下から手前に向かって出るように持ち，ロールペーパー挿入口へロールペーパーを挿入しながら，先端が2～3cm出るまでフィードキーを押してください．
（無理にペーパーを引き出さないでください）．
④ロールペーパーをボックス内に入れ，プリンタカバーをとじます．

図1

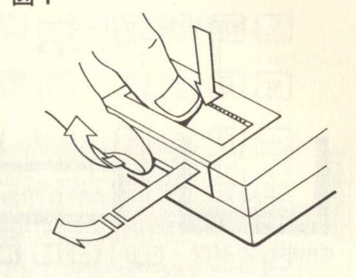


図2

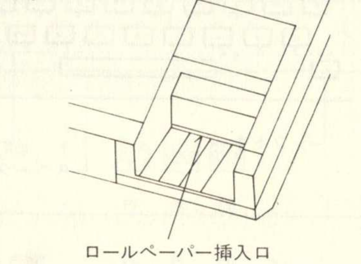
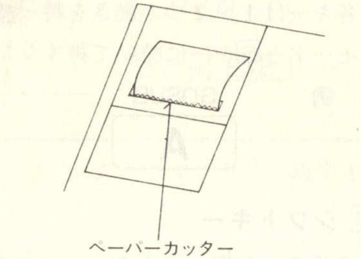


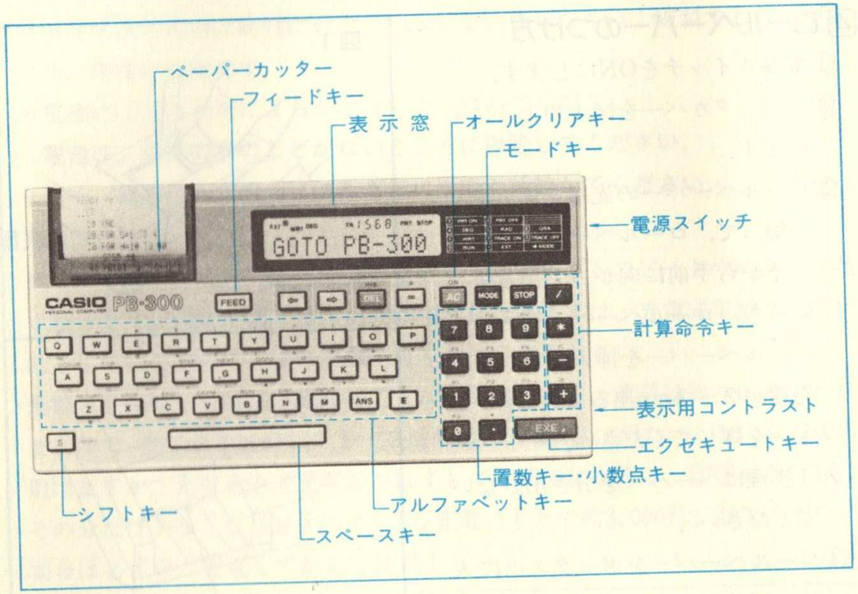
図3



お願い

本機のロールペーパーは特殊加工してありますので，一般の販売店では扱っておりません．お求めは本機お買い上げ店，またはカシオサービスセンターに「カシオPB－300用(TRP-16)」と指定してご用命ください．

2　各部の名称とそのはたらき



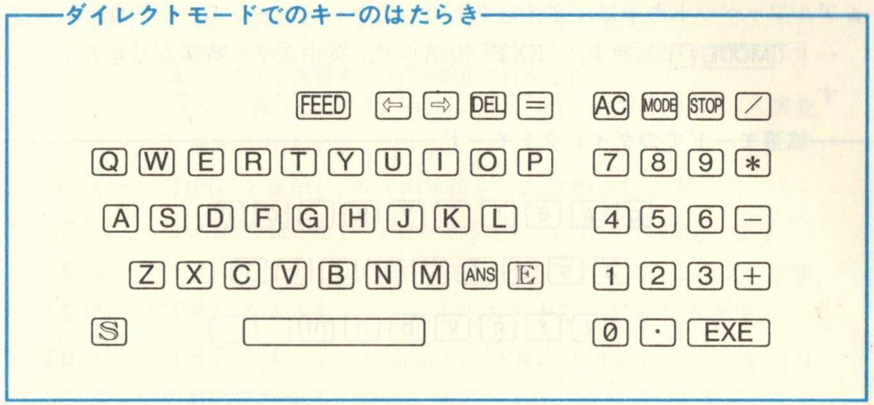
① 各部の名称

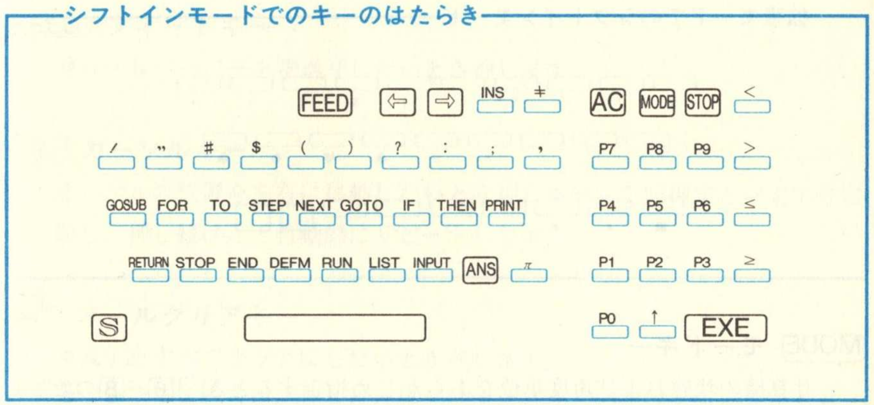
各キーは1～2つの働きを持っています．この働きは，直接押すダイレクトモードとⓈキーに続いて押すシフトインモードにより使いわけます．

例
GOSUB ―――― シフトインモード
[A] ―――― ダイレクトモード

Ⓢ シフトキー

このキーを押しますと，シフトインモードとなり（"Ⓢ"表示），キーボード上のワンキーコマンドが使えます．（以後，本書ではⓈと記します）

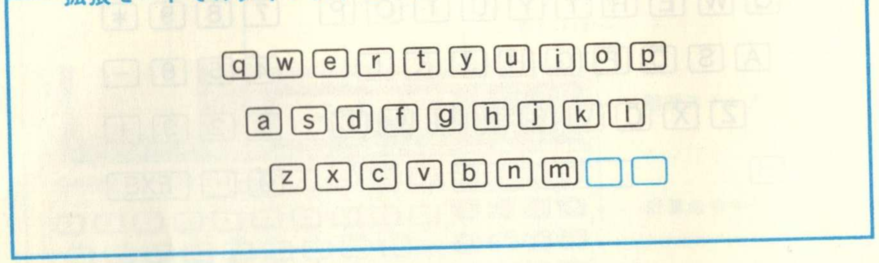




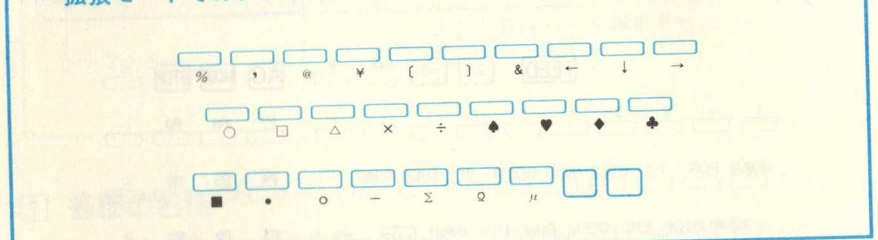
※シフトインモードではアルファベットキーはワンキーコマンドに，数字キーはプログラムエリア指定となります．

★アルファベットキーは，ダイレクトモードとシフトインモードの他に拡張モード（MODE ・ と押す．“EXT”表示）で，英小文字・特殊記号を表示します．

拡張モードでのダイレクトモード



拡張モードでのシフトインモード



MODE モードキー

計算機の状態および角度単位をあらかじめ指定するとき ・ 0 ～ 8 の数字キーと合わせて押します．

MODE ・……“EXT”を表示し，拡張モードとなり，英小文字・特殊記号が使えます．拡張モードを解除するときにも，この操作を行ないます．

MODE 0……“RUN”を表示し，マニュアル計算およびプログラム計算が行なえます．

MODE 1……“WRT”を表示し，プログラムの書き込みおよびチェック，編集が行なえます．

MODE 2……“TR”を表示し，実行トレースが行なえます．
（トレースモードでプログラムを実行すると，一行ごとに実行して止まるので，実行デバッグが簡単に行なえます）

MODE 3……“TR”が表示している場合は“TR”が消え，実行トレース機能が解除されます．

MODE 4……“DEG”を表示し，角度の単位を〈度〉に指定します．

MODE 5……“RAD”を表示し，角度の単位を〈ラジアン〉に指定します．

MODE 6……“GRA”を表示し，角度の単位を〈グラジアン〉に指定します．

MODE 7……“PRT”を表示し，プリント出力をすることができます．

MODE 8……“PRT”が表示している場合は“PRT”が消え，プリント出力が解除されます．

FEED フィードキー

●ロールペーパーを空送りしたいとき押します．

⇦⇨ カーソルキー

カーソルの位置を左右に移動したいとき押します．1回押すと一文字分移動し，押し続けると自動的にリピートします．

ON AC オールクリアキー

●表示をすべてクリアにしたいとき押します．

●プログラム実行中に押された場合はプログラムの実行を中止させます．

●エラーメッセージが表示された場合は，エラーを解除するために押します．

●オートパワーオフ（自動節電機能…26ページ参照）が働いて，表示が消えているときに電源をオンにするために押します．

INS DEL デリート/インサートキー

●カーソルが点滅している所の入力文字を1文字分削除します．

●シフトインモードでは，挿入のため1文字分あけるために押します．

STOP ストップキー

プログラム実行中に押しますと，“STOP”を表示し，一行の区切りでプログラムの実行を停止させます．
つづいて押すと，プログラムエリアNo.と行番号を表示します．

EXE エクセキュートキー

- マニュアル計算の答を求めるとき，「＝」のかわりに押します．
- “WRT”モードでは，プログラムを書き込むときに，行番号ごとの計算機への書き込み（記憶）として押します．このキーを押さないと書き込まれません．
- “RUN”モードでは，プログラムの実行中のデータ入力のためや，“STOP”表示中にプログラムの実行を継続するために押します．

ANS アンサーキー

直前に行なわれた計算結果（答）を呼び戻したいとき押します．

E（π） エキスポーネント（指数部置数）/パイキー

- 指数部を置数するとき，仮数部の置数後に押します．
  例　2.56×10^{34} ⟶ 2.56 E 34
  ※指数部は最高±99までで，それ以上はエラーとなります．

＝（≠） イコールキー/比較キー

- 代入文のときやIF文中の比較（等号）のとき押します．
- シフトインモードでは，IF文中の比較のとき押します．

7 8 9 / 4 5 6 / 1 2 3 / 0 ・（P0〜P9, ↑） 置数キー/プログラムNo.キー

- 数値を計算機に入力する（置数する）とき押します．・は小数点の位置で押します．
- シフトインモードでは 0（P0）〜 9（P9）はプログラムNo.の指定となり，プログラムが書き込まれているときはプログラムがスタートされます．
- ・（↑）キーは，シフトインモードではべき乗（x^y）を求めるとき押します．

＋（≧） −（≦） ＊（＞） /（＜） 計算命令キー/比較キー

- 四則演算をするとき，それぞれの位置で押します．
  ※＊は“×”（乗算）のとき，/は“÷”（除算）のとき押します．
- シフトインモードでは，IF文中の判断のための比較として書くとき押します．

'	"	#	$	(	)	?	:	;	,
Q	W	E	R	T	Y	U	I	O	P
GOSUB	FOR	TO	STEP	NEXT	GOTO	IF	THEN	PRINT	
A	S	D	F	G	H	J	K	L	
RETURN	STOP	END	DEFM	RUN	LIST	INPUT			
Z	X	C	V	B	N	M			

［スペースキー］

アルファベットキー/ワンキーコマンド・キャラクターキー

- プログラムを書き込むときや，コマンド・関数命令を書くとき，これらのキーを押せばアルファベットが表示されます．［スペースキー］はスペース（空白）を開けるとき押します．
- Q（'）〜 P（,）は，シフトインモードでキーパネル上に書かれているキャラクターが表示されます．
- A（GOSUB）〜 M（INPUT）は，シフトインモードでキーパネル上に書かれているワンキーコマンドを使うとき押します．

② 表示の見方



計算数値や答を表示します．それぞれの桁が横5×縦7ドットで構成され，数値や文字を最大12桁まで表示します．（数字0は0と表示します）式や文が12桁をこえた場合は数字や文字が左に送られ，最大62文字が入力できます．カーソルは入力文字が55文字までは“＿”の点滅となり，56文字以上になると“▮”の点滅となります．

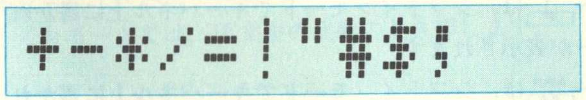
表示上部の4桁は残りのステップ数を表示します．

なお，演算中は上部4桁の一番右に“＿”が，また，“DEG”“RAD”“GRA”の角度単位や“S”（Sキーを押したとき），“RUN”(RUNモード)，“WRT”(WRTモード)，“TR”（トレースモード），“PRT”(PRTモード)，“STOP”などの記号も，それぞれの状態のとき表示します．

●**アルファベット表示例**

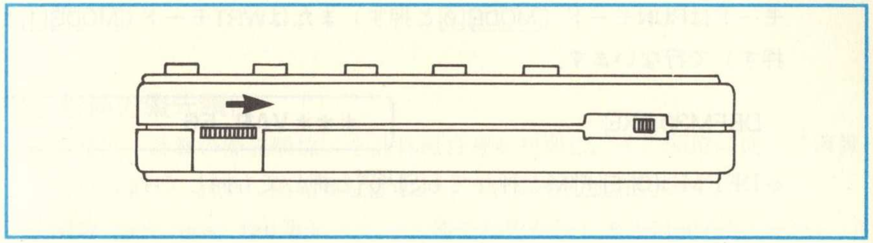


●**記号表示例**



③ コントラスト調整

表示のコントラスト（濃度）の調整は，本体右側にある調整用ボリウムで行ないます．



矢印方向に回すと表示が濃くなり，反対に回すと薄くなります．これは，電池容量による表示の濃淡の調整や，角度による見易さを調整します．

④ メモリー(変数)の増設

メモリー（変数）は通常26メモリーで，そのときのステップ数は1,568ステップとなります．

このメモリーは，最大222個まで増設することができます．このメモリーの増設は，1メモリーにつき8ステップの換算でプログラムステップをメモリーに変換します．

メモリー数	プログラムステップ数
26個	1568ステップ
27個	1560ステップ
28個	1552ステップ
〜	〜
46個	1408ステップ
〜	〜
94個	1024ステップ
〜	〜
200個	176ステップ
〜	〜
222個	0ステップ

メモリーの増設は1個単位で，DEFMコマンドを使い行ないます．

【 例 】

30個を増設して56個とします．

【操作】

モードはRUNモード（MODE 0と押す）またはWRTモード（MODE 1と押す）で行ないます．

DEFM30 EXE　　　***VAR:56

※DEFMはD E F Mと押してもS $\overset{DEFM}{V}$と押しても同じです．

現在メモリー数がいくつに設定されているかを確認するときもDEFMコマンドを使います．

【 例 】

メモリー数が合計56個に設定されています．

DEFM EXE　　　***VAR:56

★すでに相当数のプログラムステップが使用されている場合には，書き込みずみのプログラムを保護するために，ステップ数不足となる指定はエラー（ERR 1…ステップ数不足）となります．

★専用文字変数($)は，特別メモリーのため，指定時には数えません．

5 オートパワーオフ(自動電源OFF)

スイッチの切り忘れによるムダな電力消費を防ぐ自動節電機能で，操作完了後（プログラム計算中を除く）約7分で自動的に電源オフになります．

この場合は，電源スイッチを入れ直すか，AC キーを押せば，再び電源オンになります．

※電源オフになってもメモリー内容およびプログラム内容は消えませんが，角度指定や各モード指定（"WRT"，"TR"，"PRT"等）はすべて解除されます．

3 計算をはじめる前に

マニュアル計算およびプログラム計算の実行は"RUN"モード（"RUN"表示＝MODE 0）で行なってください．

なお，"DEG" "RAD" "GRA"の角度単位は，角度に関係ない計算の場合は何が表示されていても関係ありません．

1 計算の優先順位

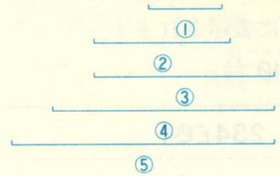
● 本機は「計算の優先順位」を計算機自身が判別し，その順位に従って演算を行ないます．計算の優先順位は次のように定められています．

①関数（sin，cos，tan等）
②べき乗
③×，÷（*，/）
④+，−

〔優先順位が同じときは頭から(左から)計算します．なお，カッコが使用されたときは，カッコ内が最優先されます．〕

【 例 】

2+3*SIN(17+13)↑2=2.75



2 入出力桁数と演算桁数

本機の入力桁数は仮数部12桁，指数部2桁で，内部演算も仮数部12桁，指数部2桁で行なわれます．

この範囲は1×10^{-99}～$\pm9.99999999999\times10^{+99}$および0です．

出力桁数は仮数部10桁ですが，指数部が付く場合は仮数部8桁，指数部2桁となります．

関数の答等で，表示桁数（12桁）をこえる場合は0と小数点および負符号を含めて12桁まで表示されます．

【例】

$(1\times10^5)\div7=14285.71429$

$(1\times10^5)\div7-14285=0.7142857$

【操作】

操作	表示
1 [E] 5 [／] 7 [EXE]	14285.71429
1 [E] 5 [／] 7 [−] 14285 [EXE]	0.7142857

計算の結果(答)が10^{10}(100億)以上または10^{-3}未満のときは，自動的に指数表示になります．

【例】

$1234567890\times10=12345678900$　$(=1.23456789\times10^{10})$

【操作】

操作	表示
1234567890 [*] 10 [EXE]	1.2345678 E10 (↑指数サイン)

※指数部は，仮数部の後に指数サインと共に表示されます．

$1.234\div10000=0.0001234$　$(=1.234\times10^{-4})$

操作	表示
1.234 [／] 10000 [EXE]	1.234 E⁻04

③ 基本的な計算の仕方

(1)演算符号と関数命令

BASICに使われる演算符号のうち，四則計算用では "+" と "−" は一般の計算式に使われるものと同じですが， "×" と "÷" は "*" と "／" を使います．

つまり，　2+3−4×5÷6

という計算式は　2+3−4*5／6

となります．

本機で使える組み込み関数は次の通りです．

関数名		書式
三角関数	sin x	SIN x
	cos x	COS x
	tan x	TAN x
逆三角関数	$\sin^{-1} x$	ASN x
	$\cos^{-1} x$	ACS x
	$\tan^{-1} x$	ATN x
平方根	$\sqrt{x}$	SQR x
指数関数	e	EXP 1★
自然対数	ln x	LN x
常用対数	log x	LOG x
整数化	INT x	INT x
整数部除去	FRAC x	FRAC x
絶対値化	$\|x\|$	ABS x
符号化	正数→1 0→0 負数→−1	SGN x
四捨五入	〔xの10^yを四捨五入〕	RND(x, y)※
乱数		RAN #

※RND関数は必ず引数を(　)でくくります．

★EXP(e)は指数表の数値を呼び出す命令です．

(2)直前の計算結果の呼び出し

マニュアル計算およびプログラム計算実行後の計算結果は，次の計算が実行されるまで記憶されています．この結果は[ANS]キーにより表示されます．

【例】

741+852=1593

2431−1593=838

【操作】

操作	表示
[7][4][1][+][8][5][2]	741+852

EXE	1593
2 4 3 1 − ANS	2431−1593
EXE	838

また，計算直後に表示された数値は，そのまま次の計算にも使えます．

【 例 】

838×2=1676

上記に続けて

【操作】

* 2	838*2
EXE	1676

(3)エラーメッセージ

数式や代入文が，BASIC言語の記法上正しくないときや，計算機が計算範囲をこえて使用されると，実行時にエラーとなり，エラーメッセージが表示されます．べき乗（$x↑y$）計算の場合は$x<0$でもyが自然数であればエラーメッセージは表示されません．

このエラーメッセージはマニュアル計算中は

ERR2	（構文エラー）
ERR3	（数学的エラー）

と表示され，プログラム計算中は

ERR2　P0−10	P0 のプログラムエリアの10行目に構文エラー発生
ERR3　P2−20	P2 のプログラムエリアの20行目に数学的エラー発生

と表示されます．

（エラーメッセージの意味は70ページを参照してください）

※演算範囲が±9.999999999E+99を越えた場合はオーバーフローとなり，エラーメッセージが表示されます．また，1.0E−99未満のときはアンダーフローとなり，演算結果は0となります．

4 キー操作

まず，電源をONにして“READY P0”の文字が表示されれば計算機の準備ができていますので，入力OKとなります．

(1)キー入力

●アルファベット（英文字）入力

【 例 】

ABCを入力する

【操作】

A B C	ABC

【 例 】

SINを入力する

【操作】

S I N	SIN

●数字入力

【 例 】

123を入力する

【操作】

1 2 3	123

【例】

96.3を入力する

【操作】

9 6 · 3　　　96.3

● 記号入力

【例】

$#?を入力する

【操作】

S R S E S U　　　$#?

【例】

¥ΣΩを入力する

【操作】

MODE · S R S B S N　　　¥ΣΩ

MODE ·（EXTモード解除）

● 指数付数値の入力

【例】

7.896×10^15を入力する

【操作】

7 · 8 9 6 E 1 5　　　7.896E15

↑ 指数部記号

【例】

−2.369×10^−45を入力する

【操作】

− 2 · 3 6 9 E − 4 5　　　−2.369E45

(2)入力内容の変更（訂正，削除，挿入）

● 訂正

訂正は，訂正したい箇所にカーソルを移動し（⇦⇨を使う），その位置で新しく訂正したい文字や数字・記号のキーを押します．

【例】

"A$"を"B$"に訂正する　　　A$_

【操作】

カーソルを左に2文字分移動させる

⇦⇦　　　A$

Bキーを押す　　　B$

【例】

"RUN"を"LIST"に訂正する　　　RUN_

【操作】

カーソルを左に3文字分移動させる

⇦⇦⇦　　　RUN

L I S Tまたは S N(LIST)と押す　　　LIST_

● 削除

削除は，削除したい箇所にカーソルを移動させ，DELキーを押します．1回押すごとに1文字分削除され，削除された文字分だけ左につめられます．

【例】

"SIIN"と書いたものから"I"を1文字分削除する．　　　SIIN_

【操作】

カーソルを２文字分移動させる

⇦⇦

```
SIIN
```

DELを押す

```
SIN
```

【 例 】

"INPUT X, Y"から"X, "を削除する．

```
INPUT X , Y_
```

【操作】

カーソルを3文字分左に移動させる

⇦⇦⇦

```
INPUT X , Y
```

DEL DELと押して2文字分削除する

```
INPUT Y
```

●**挿入**

挿入は，挿入したい箇所の右の文字にカーソルを移動させ，その位置でS INS/DEL と押せば１文字分あきますので，その箇所に挿入したい文字や数字・記号のキーを押します．

【 例 】

"T＝A$"と書いたものを

"T$＝A$"と直す．

```
T=A$_
```

【操作】

カーソルを左に3文字分移動させる

⇦⇦⇦

```
T=A$
```

S INS/DEL と押して，１文字分あける

```
T_=A$
```

挿入したい文字S $/R と押す

```
T$=A$
```


【 例 】

"PRINT X"を"PRINT SIN X"に直す

```
PRINT X_
```

【操作】

カーソルを左に１文字分移動させます

⇦

```
PRINT X
```

"SIN"を入れるために3文字分あける

S INS/DEL S INS/DEL S INS/DEL

```
PRINT_   X
```

S I Nと押す

```
PRINT SINX
```

以上が入力内容の変更の方法です．

4 マニュアル計算

① マニュアル計算とは

プログラムとして計算式を記憶させて，計算機に自動的に計算させるのではなく，数式の右辺の計算を左辺に代入したり，計算式を手計算により行なったり，変数の内容を呼び出したりする操作を行なうことをマニュアル計算と呼びます．

② マニュアル計算の操作方法

◆四則計算は完全数式通りに，+ − *（×）/（÷）とEXE（=）を使います．EXEキーは　計算結果を求めるキーで，"="の働きをします．

【例】

12+36−9×5÷4=36.75

【操作】

1 2 + 3 6 − 9 * 5 / 4	12+36−9*5/4
EXE	36.75

◆関数計算は四則計算を含めて完全数式通りで，関数コマンドの後にデータを書き込みます．

【例】

log 1.23=0.0899051114

【操作】

LOG 1.23	LOG1.23
EXE	0.0899051114

※アルファベットと数字はこの説明書では枠をはずして記します．

例：S I N S (1 5 + 8 S) EXE →SIN S (15 + 8 S) EXE

◆メモリー計算などのように，数値や計算結果を記憶させたり，合計をとる場合には変数を使います．この変数は英文字（A, B, 〜, Z）または英文字と数字（0, 1, 2, 〜9）を組み合わせたもの（配列として用いた場合）を使い，記憶用のメモリーの働きをします．

変数に数値や計算結果を入れるには，代入式を使います．

【例】

1234を変数Aに記憶させる．

【操作】

A = 1234	A=1234
EXE	_

【例】

23×56の答を変数Kに加える

【操作】

K = K + 23 * 56	K=K+23*56
EXE	_

この方法は，プログラム中の代入文と同じ操作をマニュアルで行なう方法です．

●EXEキーを押す前の訂正はカーソルを訂正箇所に合わせ，正しいキーを押します．

（33ページ参照）

●表示全部を消す場合はACを押します．

③ マニュアル計算例

基本計算の仕方

◆加減乗除計算

【例】

23＋4.5−53＝−25.5

【操作】

23[+]4.5[−]53[EXE]	−25.5

【例】

56×(−12)÷(−2.5)＝268.8

【操作】

56[＊][S][(][−]12[S][)][／][S][(][−]2.5[S][)][EXE] （省略可）	268.8

【例】

12369×7532×74103＝6.9036806×10^{12}

（＝6903680613000）

【操作】

12369[＊]7532[＊]74103[EXE]	6.9036806E 12

【例】

1.23÷90÷45.6＝2.9970760×10^{-4}

（＝0.00029970760）

【操作】

1.23[／]90[／]45.6[EXE]	2.9970760E-04

※答が10^{10}(100億)以上および10^{-3}(0.001)未満のときは指数表示となります。

【例】

7×8＋4×5＝76

【操作】

7[＊]8[+]4[＊]5[EXE]	76

【例】

12＋(2.4×10^5)÷42.6−78×36.9＝2767.602817

【操作】

12[+]2.4[E]5[／]42.6[−]78[＊]36.9[EXE]	2767.602817

◆メモリー計算

【例】

12×45＝540

12×31＝372

75÷12＝6.25

【操作】

A[=]12[EXE]	_
A[＊]45[EXE]	540
A[＊]31[EXE]	372
75[／]A[EXE]	6.25

【例】

23＋9＝32
53－6＝47
－)45×2＝90
99÷3＝33

合計　22

【操作】

M [=] 23 [+] 9 [EXE]

M [=] M [+] 53 [-] 6 [EXE]

M [=] M [-] 45 [*] 2 [EXE]

M [=] M [+] 99 [/] 3 [EXE]

M [EXE]　　　　22

※この操作方法ですと各々の計算結果がわかりませんので，計算結果を見たいときは次の方法で行ないます。

23 [+] 9 [EXE]　　　　32

M [=] [ANS] [EXE]

53 [-] 6 [EXE]　　　　47

M [=] M [+] [ANS] [EXE]

45 [*] 2 [EXE]　　　　90

M [=] M [-] [ANS] [EXE]

99 [/] 3 [EXE]　　　　33

M [=] M [+] [ANS] [EXE]

M [EXE]　　　　22

関数計算の仕方

◆三角関数(sin, cos, tan)，逆三角関数($\sin^{-1}$, $\cos^{-1}$, $\tan^{-1}$)

●三角・逆三角関数を使うときは必ず角度単位の指定を行なってください。
（角度単位を変更しない場合は新たに行なう必要はありません）

> EXP(e)は連続計算の中ではご使用になれません。
> また，マルチステートメント文でもご使用になれません。

【例】

sin12.3456°＝0.2138079201

【操作】

[MODE] [4] →"DEG"

SIN 12.3456 [EXE]　　　　DEG 0.2138079201

【例】

2·sin45°×cos65.1°＝0.5954345575

【操作】

2 [*] SIN45 [*] COS65.1 [EXE]　　　　DEG 0.5954345575

【例】

$\sin^{-1}0.5=30°$（$\sin x°=0.5$の xを求める）

【操作】

ASN0.5 EXE

DEG
30

【例】

$2.5\times(\sin^{-1}0.8-\cos^{-1}0.9)=68.22042398$

【操作】

2.5 * S (ASN0.8 − ACS0.9 S) EXE

DEG
68.22042398

【例】

$\cos(\frac{\pi}{3}\text{rad})=0.5$

【操作】

MODE 5 →“RAD”

COS S (S π / 3 S) EXE

RAD
0.5

【例】

$\cos^{-1}\frac{\sqrt{2}}{2}=0.7853981634$

【操作】

ACS S (SQR2 / 2 S) EXE

RAD
0.7853981634

【例】

tan(−35gra)=−0.612800788

【操作】

MODE 6 →“GRA”

TAN − 35 EXE

GRA
−0.612800788

◆対数関数(log, ln)，指数関数(e, x^y)

【例】

$\log 1.23(=\log_{10}1.23)=0.0899051114$

【操作】

LOG1.23 EXE

0.0899051114

【例】

$\ln 90(=\log_e 90)=4.49980967$

【操作】

LN90 EXE

4.49980967

【例】

$\log 456\div\ln 456=0.4342944819$

【操作】

LOG456 / LN456 EXE

0.4342944819

【 例 】

$e=2.718281828$

（この関数は指数表の数値を呼び出す命令です）

【操作】

EXP1 EXE　　**2.718281828**

【 例 】

$10^{1.23}=16.98243652$

（常用対数1.23の真数を求める）

【操作】

10 S ⬆ 1.23 EXE　　**16.98243652**

【 例 】

$5.6^{2.3}=52.58143837$

【操作】

5·6 S ⬆ 2.3 EXE　　**52.58143837**

【 例 】

$123^{\frac{1}{7}}(=\sqrt[7]{123})=1.988647795$

【操作】

123 S ⬆ S (1 ⁄ 7 S) EXE　　**1.988647795**

【 例 】

$\log \sin 40^\circ + \log \cos 35^\circ = -0.2785679838$

その真数は……0.5265407845（$\sin 40^\circ \times \cos 35^\circ$ の対数計算）

【操作】

MODE 4 （DEG指定）

LOG SIN 40 + LOG COS 35 EXE　　**−0.278567983**

10 S ⬆ ANS EXE　　**0.5265407845**

※べき乗（$x\uparrow y$）計算のxの値は$x>0$です．

◆その他の関数（√‾, SGN, RAN#, RND, ABS, INT, FRAC）

【 例 】

$\sqrt{2}+\sqrt{5}=3.65028154$

【操作】

SQR2 + SQR5 EXE　　**3.65028154**

【 例 】

正数であれば“１”を，負数であれば“－１”を，0であれば“0”を与える

【操作】

SGN6 EXE　　**1**

SGN0 EXE　　**0**

SGN − 2 EXE　　**−1**

【例】

乱数発生（∅<RAN#<1の疑似乱数）

【操作】

RAN S # EXE　　**0.904186914**

【例】

12.3×4.56の答を10^{-2}の位（小数点以下第2位）で四捨五入する

12.3×4.56=56.088

【操作】

RND S (12.3 * 4.56 S , −2 S) EXE

56.1

【例】

｜−78.9÷5.6｜=14.08928571

【操作】

ABS S (−78.9 / 5.6 S) EXE　　**14.08928571**

【例】

$\frac{7800}{96}$の整数部は………81

【操作】

INT S (7800 / 96 S) EXE　　**81**

※この命令は元の数値を超えない最大の整数を求めます．

【例】

$\frac{7800}{96}$の小数部は………0.25

【操作】

FRAC S (7800 / 96 S) EXE　　**0.25**

◆有効桁数指定，小数以下指定

有効桁数と小数以下の指定は“SET”コマンドにより行ないます．

有効桁数指定……SET E *n*　（*n*=0～8）

小数以下指定……SET F *n*　（*n*=0～9）

指定解除…………SET N

※有効桁数指定の場合の“SET E∅”は8桁指定となります．

※指定を行なうと，指定桁の下1桁目を四捨五入して表示します．なお，計算機内部やメモリー内にはもとの数値が残っています．

【例】

100÷6=16.66666666……

【操作】

SET E4 EXE（有効桁数4桁指定）

100 / 6 EXE　　**1.667E01**

【例】

123÷7=17.57142857……

【操作】

SET F2 EXE（小数以下2桁指定）

123 ⁄ 7 EXE　　　　17.57

【 例 】

1÷3＝0.3333333333……

【操作】

SET N EXE（指定解除）

1 ⁄ 3 EXE　　　　0.3333333333

5 配　列

◆配　列

配列は一次元配列が使え，A(*i*) とかB(*j*) のように添字が付いた形で使われます．

この配列は通常の26メモリーとDEFMコマンドにより増設されたメモリーを使いますので，次の配列の取り方に注意してください．

A＝A(0)

B＝A(1)＝B(0)

C＝A(2)＝B(1)＝C(0)

D＝A(3)＝B(2)＝C(1)＝D(0)

E＝A(4)＝B(3)＝C(2)＝D(1)＝E(0)

⋮

Y＝A(24)＝B(23)＝C(22) ……………… ＝Y(0)

Z＝A(25)＝B(24)＝ ……………………… ＝Y(1)＝Z(0)

増設メモリー
A(26)＝B(25)＝ ……………………… ＝Y(2)＝Z(1)
A(27)＝B(26)＝ ……………………… ＝Y(3)＝Z(2)
⋮
A(221)＝B(220)＝ …………………… ＝Y(197)＝Z(196)

このように配列を使う場合は，配列の引数により同じメモリーになることがありますので，同一プログラム中で同じメモリーを使わないようにしてください．

【 例 】

同時に使える………A,B,C,F(0), F(9)

同時に使えない………F,G,A(5), A(6)

なお，配列の大きさによってメモリーの増設は正確に行なってください．

第2編

BASICプログラミング編

1　PB-300を大いに使おう

① 身近なパソコン

PB－300は小型に作られたパソコンだから，さぞや中身はギッシリつまっていて，取り扱いもデリケートなのではないか，そんな心配をまずされるかもしれません．

LSI関連の技術は，日進月歩，日々革新されています．大容量，多機能のLSIが，とても小さく作れるまでに進んでいます．ICやトランジスタ，抵抗，コンデンサなどを，何個も寄せ集めて作られていた回路が，1個のLSI，それも2センチ角程度のプラスチックパッケージの中に納められてしまうと，小型化が可能なのはもちろん，故障することも，グーンと少なくなりました．

こうした技術に裏付けされて，パソコンの小型化，少部品化，省電力，高性能，高機能が実現しました．

PB－300の中は，意外に思われるほどスッキリまとまっています．使われている部品点数が少ないのです．

手に持ってください．軽いですね．重さは258グラム（電池込み）です．

コンパクトに作られて軽いことは，手軽に持ち運べることを意味します．

カバンの中やポケットに入れたとしても，全く負担になりません．だから，どこへでも持って行ってください．電車の中でも，会社や学校でも，使う場所に制限されずにプログラムが作れますし，問題解決に役立ちます．

みなさんが一旦，プログラムの勉強を始めると，“あの命令はどうなっていたっけな”なんて疑問が，道を歩いているときや，仕事，勉強中にだって，時と場所を選ばず浮かんできます．授業中や勤務中に，全く関係のないパソコンを使うことは考えものですが，他に迷惑をかけない状況でなら，カバンやポケットから，サッとPB－300を取り出して試してください．PB－300が理解できるにつれて，家に帰ってから，なんて待っていられなくなるのが本音です．

このようにして使っていくうちに，いつのまにか，PB－300とプログラミング言語のBASICを自由に使いこなせるようになります．

もちろん，PB－300を即，業務に使おうと考えている人も同じです．

問題が発生したときに，PB－300が手元にあれば，いつでも問題解決の道具



として使えます．業務は机の上だけで処理できるものとは限りません．問題や仕事のある場所で，PB－300を使えばよいのですから，これは最適です．

いつも使い慣れたPB－300を身近に持っていることが，パソコン活用の第1歩です．

② ミニプリンタも付いて，大活躍

パソコンに限らず，コンピュータ全般の代表的な出力装置として，プリンタが上げられます．

プリンタは，たとえて言うならば，荷物を手に持って運んでいたのに対し，トラックを使い，何倍も荷物を，短時間で運べる，といったパソコンの機動力と有効性の向上を実現する装置です．

プリンタの利用方法を考えてみますと，データの印字，グラフや表の作成，

■異常な使い方はしないでください■

PB－300は，持ち運んで大いに使っていただきたいのですが，ほうり投げたり，水をかける，火に入れるなどの異常な使い方をしないでください．

このような使い方をしない限り，故障する可能性はほとんどありません．

また，コネクタに電線を差し込んでショートさせたり，裏ぶたをはずしたついでに，プラスチックカバーまではずして，中を触ったりしないようにしてください．LSIは，静電気に弱いので，人体に蓄えられている静電気で壊れる可能性があります．コネクタをショートすれば，電池のショートやLSIの端子をショートして，故障の原因となります．

くれぐれも，異常な使い方は避けてください．そうすれば，いつまでも愛用機として，実用パソコンとして活用できます．



記録や保存，表示装置だけにとらわれない新しい形のゲーム，図形を描く，プログラムリストの印字など，多くの分野，多くの種類の印字に活躍します．

PB−300には，プリンタが本体と一体化され，コンパクトにまとめられています．パソコンの応用範囲を広げるプリンタが付いているのですから，大いに活用してください．

ディスプレイでは必要な件数のデータを表示するのに対し，プリンタは縦・横2方向への広がりを見ることができます．データを印字すれば，前後関係や時系列の変化が1件1件の記憶によらず，記録として確かめられるわけです．

数値データであれば，グラフ化するという手もあります．数値で説得し，グラフで納得させるという，基本的な形の有効な資料が手軽に作れます．

外出するときは，データ入力用のプログラムをPB−300へ記憶させます．持って歩くにもコンパクトなので負担になりません．データの発生した場所で，その都度入力します．電源を切っても，データが消えない機能が備わっているので安心です．

業務によっては，データの入力と同時に，計算と伝票発行が必要になるでしょう．このような場合も，プリンタの付いているPB−300なら処理が可能です．

ノートや伝票に書いておいたデータを，まとめて入力するよりも，正確に，また短時間に処理できます．

プリンタは，プログラム作成時にも利用できます．

短いプログラムなら，それほど負担にならないデバッグ（プログラムの誤りの修正）も，プログラムが長くなるに従って，なかなか容易でなくなります．

そんなとき，プログラムリストを印字します．そうやって，リストを見直すと，記号や文字の間違いが，わりと簡単に発見できます．

変数の使い方や変化も，リストを見ながら実行順に追うことができます．

トレースモードと合わせて活用すれば，デバッグもスムーズに進むでしょう．

プログラムの入力から完成までの時間も，グーンと短縮できます．

プログラムのリストは，デバッグのときだけでなく，プログラムが完成したときにも印字しておきます．

ノートを一冊用意して，プログラムのリストを張り付けます．プログラムの名称，内容，入力するデータ，またはキー，出力結果，変数名，変数エリアの拡張数，プログラム作成日付など，後日プログラムを見直すとき必要な項目を合わせて記入しておきます．

こうしておけば，プログラムライブラリー集が出来上がります．このライブラリー集は，とても楽しいプログラムのアルバムとなるでしょう．

いつごろ，どんなプログラムを作ったのか，後日見直すのも楽しいですし，整理されているので，これまでのプログラムを応用した，新たなプログラムを作るのも容易です．

利用価値の高いプログラムが出来上がったら，プログラムのリストと必要な項目をまとめて，雑誌などへ発表するのもいいですね．

プリンタが一体化されていることで，PB−300 の応用範囲がずい分広がることがおわかりになると思います．

機動性に富むPB−300は，BASICの学習機から，ゲーム，技術計算，データ処理などの実務面まで，幅広い利用範囲をカバーします．



2 PB-300の仕組み

1 PB−300のハードウェア

PB−300は，コンピュータが一般に備えている5つの機能をもっています．

それは，入力装置，出力装置，記憶装置，演算装置と制御装置です．

これらの装置を簡単に説明しますと，次のようになります．

入力装置：キーボード

出力装置：表示部（液晶12桁），ミニプリンタ（感熱記録方式20桁）

＊こんな小さなコンピュータに，プリンタが付いているのが，PB−300の大きな特徴です．

記憶装置：

主メモリー { ROM（ロム）（Read Only Memory）／RAM（ラム）（Random Access Memory）

補助メモリー：カセットテープレコーダ

＊ROMは読み出し専用メモリーで，BASIC 言語やPB−300がスムーズに処理するための手順など，消えては困るものが入っています．また，RAMは，読み出しも，書き込みもできるメモリーで，わたしたちが，データやプログラムを記憶させたり，読み出したりします．

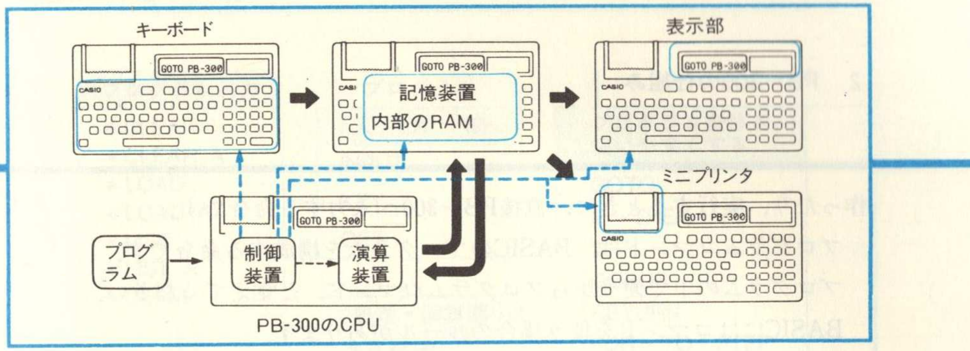
演算装置：四則計算や比較判断などをする装置です．

制御装置：入出力，記憶，演算装置などに，それぞれ受けもちの仕事をさせるために，タイミングよく指令を出す役目をします．

＊演算装置と制御装置は，コンピュータの中枢部であり，この2つの装置をまとめて特にCPU（Central Processing Unit）と呼んでいます．

これらの装置は，たとえば次のように有機的に動作します．

まず，キーボードからプログラムを入力すると，このプログラムは RAMへ記憶されます．CPUは，このプログラムに基づいて，何をすればよいかを決めます．たとえば，データの入力を要求するなら，表示部にデータ入力要求のサイン（？）を表示します．キーボードからデータを入力すると，これを RAMへ一時的に記憶します．次に計算が必要なら，RAMからこのデータをもってきて計算し，再びRAMへ返します．計算結果を表示させるプログラムなら，CPUはRAMのデータを表示部へ出力します．このように，PB−300の5つの装置はソフトウェア（プログラム）の助けを借りて，データのやり取りや計算をします．



2 ソフトウェア

PB−300は，ハードウェアとソフトウェアが合わさって本領を発揮します．

PB−300のソフトウェアには，プログラミング言語と，プログラミング言語で作ったプログラム（アプリケーションソフト）の2つがあります．

プログラミング言語として，PB−300ではBASICが使われます．

アプリケーションソフトとは，BASICで作ったゲームや計算のプログラムのことです．

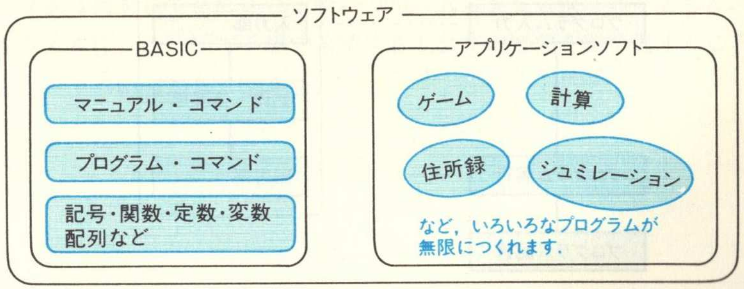
■ BASICとは

BASICは，もう耳慣れた言葉ですね．

BASICは，コマンドと記号，関数，定数，変数の2つのグループに分けることができます．

コマンドは，マニュアルコマンドとプログラムコマンドに分けられます．このコマンドのことを命令とも呼びます．PB−300へ「この仕事をしなさい」という具合に命令するわけです．

マニュアルコマンドは，BASICのプログラム（アプリケーションソフト）を



作ったり，実行するときに，直接PB－300へ入力する命令です．

プログラムコマンドは，BASICのプログラムを構成する命令です．

プログラムの中で使うからプログラムコマンド，と覚えてください．

BASICにはコマンドを使う場合のルールがあります．

このルールに違反すると，PB－300から「ERR 2」といったエラーメッセージをもらいます．

エラーメッセージが出たら，プログラムを修正します．こうして，アプリケーションソフトの出来上がりです．

3 プログラムの仕組み

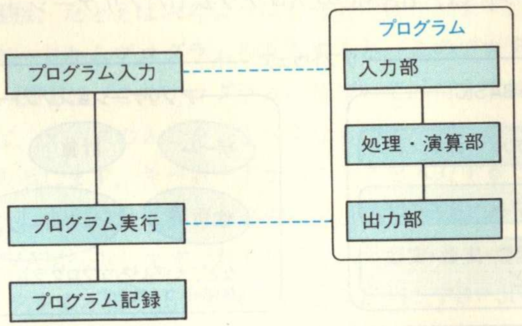
アプリケーションソフト（以下プログラムと呼びます）は，通常，3つの構成に分けることができます．

入力部，処理・演算部，出力部です．これはハードウェアの構成装置と似ています．

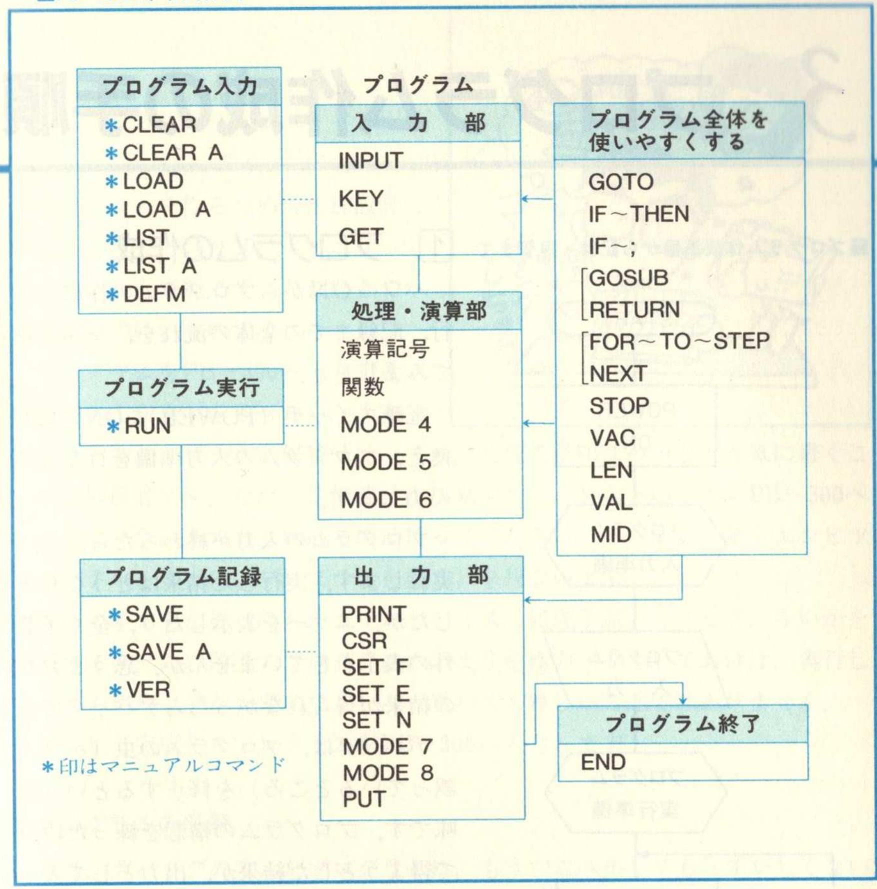
それぞれの構成ごとに，専用のコマンド（命令）がBASICには用意されています．入力部にはINPUT，出力部にはPRINTなどが代表例です．

また，処理・演算部には，多くのプログラムコマンドがあります．

一方，これらのプログラムコマンドで構成されたプログラムをキーボードから入力して，PB－300へ記憶させる作業も必要です．入力されたプログラムを実行しようとすれば，そのときもコマンドを使って実行を開始します．



■PB－300命令関係図



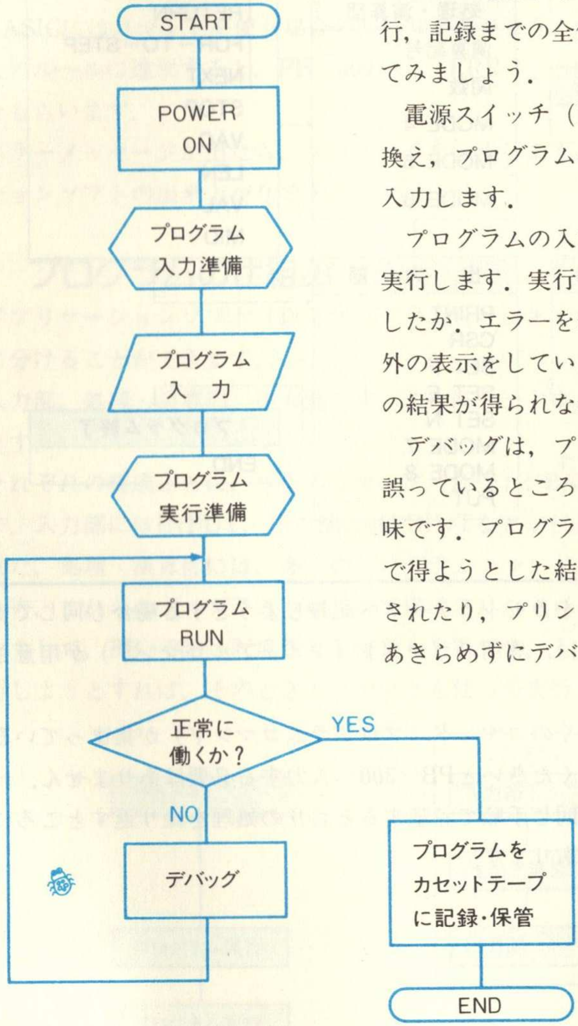
さらに，プログラムをカセットテープへ記録しようとする場合も同じです．

これらの作業のために，専用のコマンド（マニュアルコマンド）が用意されています．

プログラムでは，多くのコマンド（プログラムコマンド）が集まっているので，一々次は何をしてくださいとPB－300へ入力する必要はありません．一回プログラムを作れば，同じ手順で希望するとおりの処理を繰り返すところに，コンピュータの特長があります．

3 プログラム作成の手順

■プログラム作成準備から記録・保管まで



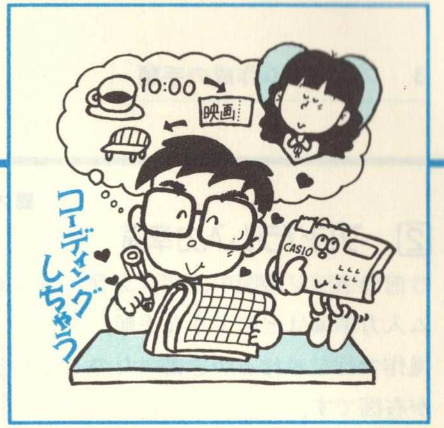
1 プログラムの作成

パワーONからプログラムの作成，実行，記録までの全体の流れを，左図で見てみましょう．

電源スイッチ（POWER)をONに切り換え，プログラムの入力準備を行なって入力します．

プログラムの入力が終わったら，次は実行します．実行した結果はどうなりましたか．エラーを表示したり，全く予想外の表示をしていませんか．思うとおりの結果が得られなかったらデバッグです．

デバッグは，プログラムの虫（バグ：誤っているところ）を修正するという意味です．プログラムの構想を練った段階で得ようとした結果が，出力として表示されたり，プリントアウトされるまで，あきらめずにデバッグしてください．



プログラムを作るための仕様設計がしっかりしていれば，ノーエラーで1発パスも可能です．これはとても気持ちのいいものです．

プログラムをPB－300へ入力する前に，こんな表示をさせたい，入力はあれとあれ，データの保管はテープへ，処理の手順は……などを紙に書くことが仕様設計です．ただ，仕様設計が出来上がったからといって，PB－300へすぐプログラムを入力し始めるのは待ってください．その前に，紙とエンピツでコーディング（紙にプログラムを書く）を行ないます．

コーディングが終ったら，コマンドのミス，処理手順のミスなどがないかを確かめます．この段階でプログラムの誤まりを極力少なくしておけば，実行してからのデバッグが大変楽ですし，一発パスも難しいことではありません．

こうして完成したプログラムをPB－300へ入力します．

■プログラムの保管

さて，デバッグのすんだプログラムは，もうどこへ出しても恥ずかしくない，立派な作品です．ミニプリンタが一体化されているPB－300では，すぐにプログラムリストをプリントアウトしてください．

プログラムリストは，記録として残りますし，次の新しいプログラムを作るときの参考資料となります．プログラムの中で良いテクニックがあれば，それは次のプログラムに応用できるというわけです．

ステップ数の大きいプログラムを入れたままでは，PB－300はその1本のプログラムの専用機になってしまいます．そこで，プログラムをキーから再入力する，という方法を使わずに，手軽に保管できるのがカセットテープです．完成したプログラムは，カセットインタフェイス（FA－3）を使いカセットテープへ記録しておけば，いつでも，テープから読み出すことができます．もちろん，エラーが発生する未完のプログラムでも，記録，読み出しはできます．

こうして完成したプログラムは，いつでも使える，あなたの財産となります．

② プログラム入力準備

前のページの図中，プログラム入力準備はどのような手順，操作で行なわれるかを書いたのが右図です．

電源スイッチをONした状態では，①RUNモード，②DEG，③プログラムエリアＰ０，に設定されています．

MODE 1 でWRT（ライト）モードに切り換えます．これでプログラムの入力は可能ですが，もう少しやっておくべきことがあります．

S（SHIFT）0から9で，プログラムをどのエリアに書くかを指定します．

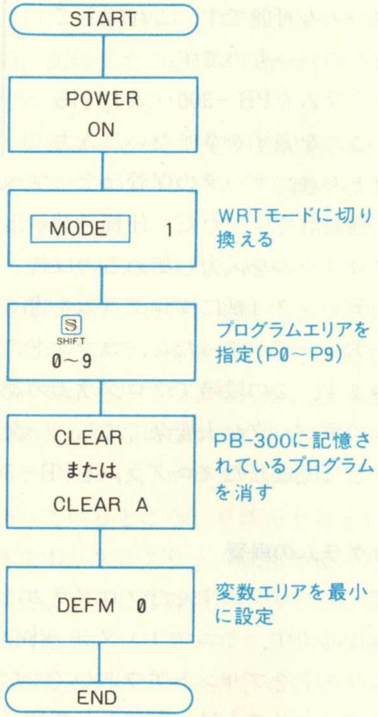
電源ON直後で，プログラムエリアの指定を省略して，プログラムの入力を始めたら，Ｐ０に記憶されます．CLEARまたはCLEAR A というマニュアルコマンドを実行して，指定されているプログラムエリア，またはすべてのプログラムエリアに記憶されているプログラムを消します．

このコマンドを実行すると，泣いても，たたいても，消えたプログラムは２度とメモリー上に現われません．使うときは十分注意してください．

DEFM 0 は，マニュアルコマンドです．変数エリアを拡張なしの基本26個に設定します．

こうして，のびのびとプログラムを入力するための準備が終わりました．

■プログラムの入力準備



③ プログラムの入力

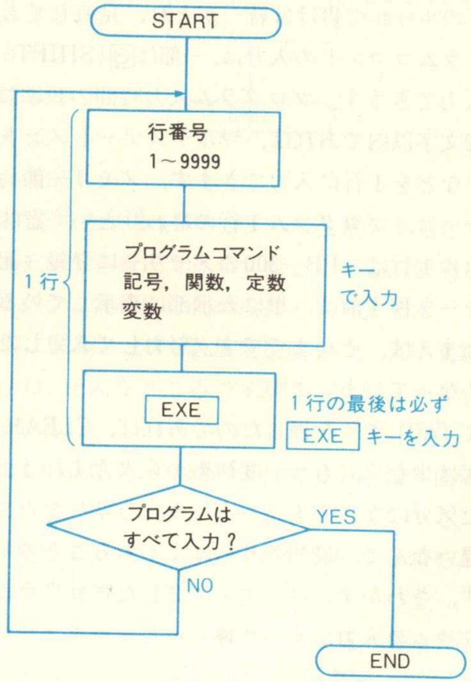
プログラムは行番号とプログラムコマンド，記号，関数，定数，変数などで構成する作業手順の集まりです．

さて，プログラムの入力準備はできているので，プログラムを入力しましょう．手順，操作を示すのが下図です．

プログラムの１行は，行番号といくつかのプログラムコマンド，記号，関数，定数，変数，配列，それに１行の終わりであるEXEキーから構成します．

１行は62文字以内とPB−300の制限があるので，紙にコーディングするとき

■プログラムの入力手順



は，文字数にも注意してください．

PB－300に入力している途中で，62文字をオーバーしそうになり，2行に分割したら行番号がメチャクチャになった，なんてことは避けなければなりません．



行番号はデバッグ時の追加，処理の追加などに対応できるように，10飛びに付けておくと便利です．

また，処理のかたまり（ブロック）ごとに，頭1文字を同じにする，などもプログラムを見やすくする工夫です．

たとえば，入力は1000番台，出力は4000番台と自分で決め，すべてのプログラムをそのルールで書けば統一がとれ，見直してもわかりやすくなります．

プログラムコマンドの入力は，一部はS（SHIFT）キーとアルファベットキーの2操作で入力できます．プログラム入力時間の短縮にとてもありがたい機能です．

1行62文字以内であれば，マルチステートメントの機能を活用して，複数のコマンドなどを1行に入力できます．メモリー節約に有効な方法です．

EXEキーは，プログラム1行の終わりという意味です．このキーを押せばプログラムの1行は，PB－300のメモリーに登録（記憶）されます．

EXEキーを押す前は，単に表示部に表示しているにすぎません．ACキーで消してしまえば，それまでです．努力して入力した行が，その行に限って，すべてなくなってしまいます．

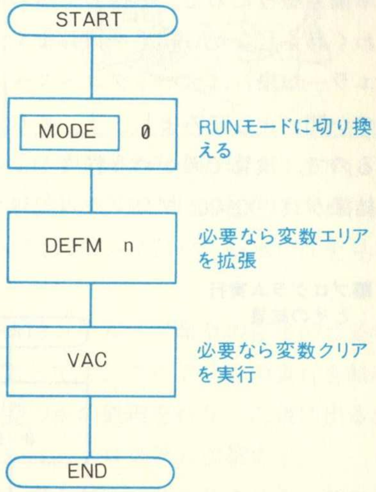
誤ってACキーを押したのであれば，CLEARを実行したことを思えば，まだ被害は少なく，もう一度初めから入力すればよいわけですから，いくらか救われた気分になれます．あまりガッカリしないことです．DELとACキーの押し間違いなんて，最初のうちはよくあることですから．

焦らず，さわがず，コーディングしたプログラムを1行ずつ入力します．

プログラムの入力がすべて終わったら，次はプログラムの実行準備です．

④ プログラムの実行準備

■プログラムの実行準備



プログラムはすべて入力したので，次ははやる気持ちを抑えて，プログラムの実行準備です．

といっても，これは多くの操作や手順を必要としません（右図）．

プログラムを入力するときは，MODE 1のWRTモードでしたが，実行するときは，MODE 0のRUNモードに切り換えます．

DEFMは，プログラムの入力準備でも実行したマニュアルコマンドです．実行準備では，変数エリアの拡張が必要なら，たとえば，DEFM10のように，拡張数を指定します．必要以上に変数エリアを拡張すると，メモリーの容量が不足して「ERR 1」を表示します．また，メモリー容量内で拡張したとしても，使用しない変数エリアをもつことは，プログラム実行には何の役にも立ちません．

VACは変数をすべて初期値に設定します．数値変数は0に，文字変数はヌルに設定されます．VACの働きは，そんなところですが，いつもプログラムの実行準備で実行するとは限りません．

PB－300の変数に記憶されている値を，今回の実行でも引き続き使いたい場合は必要ありません．もう1つは，VACはプログラムコマンドなので，プログラムの中で実行することも可能です．プログラムの中で実行するのであれば，ここでVACを実行する必要はありません．

前に，CLEARというプログラムを消すコマンドについて述べましたが，VACとよく似ていますね．プログラムと変数を消すのに，それぞれコマンドがあることを覚えておいてください．

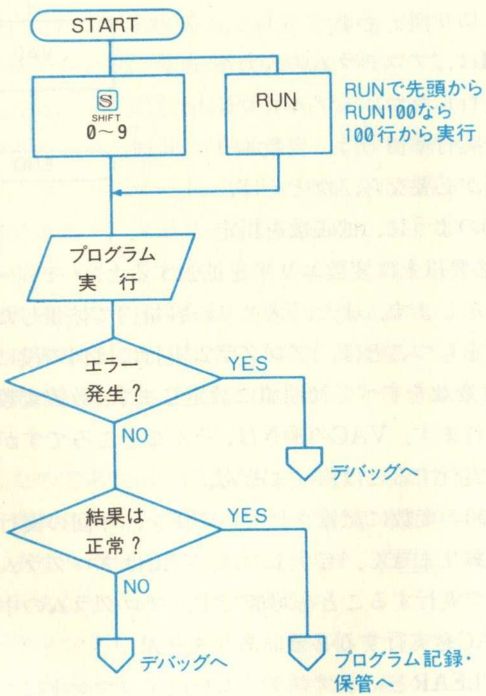
5 プログラムの実行

準備も整ったので，気持ちを落ち着けて，いよいよプログラムの実行です．わくわくしながら結果を待ちます．

エラーが出ればデバッグコースへ，結果が表示されたら，ほんとにそれで良いのか問い正してみましょう．でも，あまり深く考えると，何も信じられなくなるので，検算で確かめる程度でよいでしょう．

結果が良いなら，プログラム記録コースへ進めます．ここが分れ道です．

■プログラム実行とその結果





さて，実行の方法ですが，マニュアルコマンドRUN EXE と入力するか，プログラムを実行させたいプログラムエリアをS0～9で指定すれば，実行開始です．

手順は，前ページに示します．

S0～9で，プログラムを実行すると，必ず先頭の行から始まります．

RUN EXE でも，同じく先頭から始まりますが，RUN100 EXE とすれば，100行から実行を始めます．すでにデータの入力は終わっていて，結果だけを見たいときなどに便利です．

注意しなければならないのは，サブルーチンやループ命令の途中の行番号からRUNで指定したらエラーが発生することです．プログラムの実行を始めてすぐにエラーが発生するか，それともいくつかの処理を行なった後に出るか，はたまた，エラーなしで結果を表示するかは，プログラム次第です．

エラーは，ただ「エラーですよ」と表示するだけではありません．プログラムの実行中のエラーなら，「ERR2　P0－120」のように，エラーの発生した行番号を親切に表示してくれます．詳しい意味は次の項で説明します．

さて，このエラーが発生した場合は，プログラムが正常に実行できるようにエラーの原因を修正してやらねばなりません．エラーメッセージ（ERR2　P0－120をこう呼びます）を紙にメモして，デバッグの項に進んでください．

幸い，エラーが発生せずに結果が得られたら，その結果が正しいかどうか，確かめる必要があります．計算式であれば検算を行ないます．検算だけは面倒がらず，簡単な値でいいのでやってください．そうすれば，その後，何十年でも，このプログラムが消えない限り，楽ができます．

データの加工であれば，入力したデータと結果を比較してください．途中でデータがバケているようなことがないか，確かめます．

エラーの関門と疑いの関門を無事通り抜けたプログラムは完成品です．

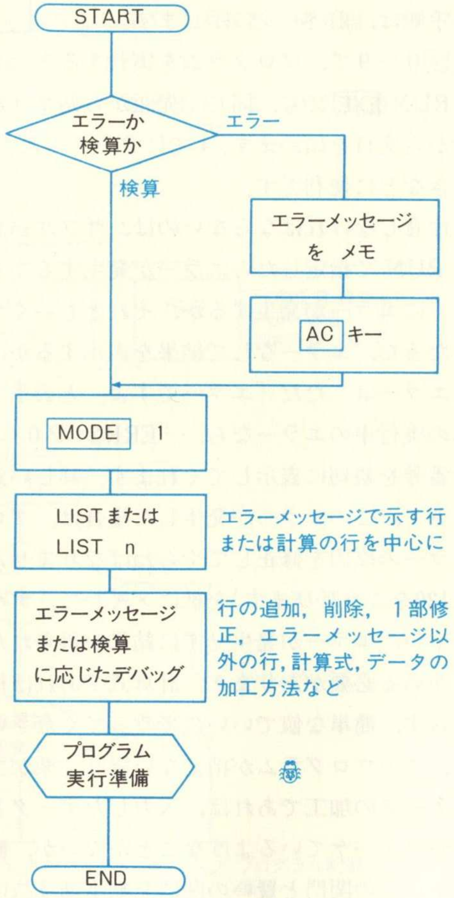
こうして完成したプログラムは，カセットテープへ記録し，保管します．

⑥ デバッグ

さて，プログラムを RUN して，エラーを表示して止まったり，結果を表示したが，満足のいくものでなかったり，結果が間違っていたら，デバッグです．

デバッグは，エラーが発生した場合と，結果が満足いかない（間違っている）場合では，やり方がちょっと違います．

■デバッグの手順



■ エラー発生の場合

エラーメッセージを表示して止まったら，まずエラーメッセージをメモします．

これがデバッグを進めるとき，犯人探しの重要証拠です．

エラーメッセージは，1例として，「ERR6 P0−100」と表示される場合と，「ERR 1」と表示される場合の2通りがあります．

エラーコードには，1から7と9が使われ，それぞれエラーの種類に応じて表示されます．



P0−100は，プログラムエリアP0の行番号100でエラーとなったことを教えてくれます．

このように，行番号を指定したエラーメッセージとそうでないメッセージの2通りがあります．

ところで，行番号が表示されたからといって，名探偵はだまされてはいけません．P0−100と表示されたとしても，その原因が必ずプログラムエリアP0の100行にあるとは限らないからです．

というのは，100行の交遊関係は，意外に広いかもしれないからです．

もちろん，100行が原因の場合もあります．エラーメッセージだけでは，どちらとも断定できませんが，重要な手掛かりには違いありません．

プログラムエリアと行番号を表示する，しないの違いですが，プログラム実行中でなければ（マニュアルモードなら），プログラムエリアがどこであろうと，エラーの発生する直前にキー入力したことが原因なので，エラーメッセージだけを表示して注意を促します．

一方，プログラム実行中に発生したエラーは，エラーメッセージに続いて，プログラムエリアと行番号を表示します．エラーメッセージの2通りとは，これだけの違いです．

ただし特例として，VAL関数だけは，プログラム実行中でも，エラーメッセージ（たとえばERR 2）しか表示しません．

エラーメッセージのわかったところで，次に進みましょう．

エラーメッセージが表示された状態では，どのキーを押してもPB−300は反応しません．じゃ，電源スイッチを切っちゃえ，ということになりますが，まア，それもよいでしょう．しかし，1つだけ受け付けてくれるキーがあります．

それが AC キーです．これを押してみてください．すると，エラーメッセージが消えて，カーソルが点滅し始めます．

これで，他のキーの入力が可能です．

特殊キーの中では，[AC]キーが一番偉そうですね．エラーを表示していても，プログラムの実行中でも，あるいはオートパワーオフの状態でも受け付けてくれるわけですから．

さて，[MODE] 1でWRTモードに切り換えます．

LIST エラーメッセージの行番号[EXE]でプログラムを表示させます．

ここからいよいよ，本格的な犯人探しの始まりです．前にも述べたように，あまりエラーメッセージの行番号にはこだわらず，あちこち見渡してください．

これが犯人探しのテクニックです．

プログラムを表示させたところで，エラーメッセージから，犯人（バグ）の似顔絵を描いてみましょう．

ERR 1なら，演算を13回以上1文でやっていることはないですか．ゴチャゴチャに書かれた計算式があったら，2つ以上に分けてスッキリ書いてみてください．

ERR 2はよく現われます．プログラムの書き方やコマンドの書き方は，BASICのルールどおりですか．数値と文字で計算したり，数値変数に文字を入力したり，代入していませんか．前の行や関係する行も調べてください．

ERR 3は計算式で発生します．演算した結果が，PB−300の演算範囲を越えていたり，負数の対数を求めようとしたり，0で割ったり，といったことが考えられます．

変数に数値を代入して計算していると，気付かないうちにERR 3の原因をつくることがあります．

ERR 4は，GOTOやGOSUBのコマンドに続く行番号が，プログラムの中に存在しない場合に発生します．これは，比較的楽に犯人が探せそうです．

GOTOやGOSUBで分岐して行く先の行番号が，見当違いにならないよう，LISTで確認してから修正してください．

ERR 5は，たとえばGOTO 100のように，引数を必要とするコマンドや関数で，引数の値がPB−300で扱える範囲外の場合に発生します．

引数を必要とするのは，RND，SQR，MID，CSR，GOTO，GOSUB，IF～THEN，MODEなどです．



ERR 6は，変数の使い方が適切でない場合に発生します．

DEFMコマンドで，拡張していない場合にもERR 6です．

1つのプログラムの中で，1つの変数名を数値変数と文字変数に使うと発生します．

たとえば，AとA$といった使い方です．

また，PRINT文で1行が62文字を越えても発生するので，PRINT命令で書かれる数，文字数も見直してください．

ERR 7は，サブルーチンとループ命令で発生するエラーです．

GOSUBは8回までの繰り返し（ネスティング）はできますが，9回目ではエラーが発生します．また，GOSUB命令ではなく，他の分岐命令でサブルーチンへ分岐した場合も，RETURNを実行したところでERR 7を表示します．

どこからサブルーチンへ分岐してきたのか，探さなくてはなりません．

FORとNEXTの間には，別の変数で書かれるFOR～NEXTが4回まで許されています．5重目のネスティングでエラーが発生します．

また，FOR～NEXTの途中へ，分岐命令で飛び込んだ場合は，NEXTを実行したところで，どこへ戻ればよいのかわからなくてエラーを発生します．

これも，どこからループの中へ飛び込んできたのか，探さなくてはいけません．

ERR 9は，カセットインタフェイスを使用する際に発生するエラーです．

カセットインタフェイスを接続しないで，SAVEやPUTを実行した場合にERR 9を表示します．

LOADやGETではPB−300は，プログラムやデータの入力を待った状態で，いつまでたってもERR 9を表示しません．

カセットインタフェイスとPB−300が接続されていたら，SAVE，PUTは通常通り動作するので，エラーは発生しません．LOADとGETは205ページをご覧ください．



以上がエラーメッセージを元に描いた犯人像です．プログラムの全体，あるいは一部の手順を見渡し，犯人（バグ）を見つけ出してください．

バグが見つかったら退治します．

計算手順を変える，新しくコマンド・演算を追加する，コマンドなどを削除するなどの大物バグの可能性もあります．

1つ1つ，つぶしていきましょう．

なお，ERR 1～7と9は，巻末の一覧表にまとめておきましたので，バグ退治には，それを利用してください．

■ 結果が間違っている，満足がいかない場合

こちらは，コマンドの使い方や計算方法はBASICの約束どおりです．

プログラムの手順や計算式，演算記号の誤り，データの受け渡し方法などを確かめてください．

試験用のデータを用意して，プログラムを1行ずつ紙の上で実行するか，PB－300のトレース機能（MODE 2）を活用して，どこで実行順序が違ってしまうのか，変数の値が変わるのかを探して，原因を追求してください．

場合によっては，プログラムを作る元になった手法，計算式が誤っていることもあります．是非，検討の対象に入れてください．

以上のような手順を踏んで，バグを退治したら，プログラムの実行準備を経て，プログラムを再度実行します．そうして，エラーが発生しなくなるまで，また，結果が満足できるまで，プログラム実行とデバッグを繰り返します．

4 マニュアルコマンド

1 CLEAR
プログラムの消去

命令	CLEAR CLEAR A	読み方：クリア ：クリアエー
書式例	例1．CLEAR 例2．CLEAR A	Aの語源：All
実行	MODE 1（WRTモード）のみで実行可能	

1.1 CLEAR ………例1

PB－300の現在指定されているプログラムエリアに，プログラムが記憶されていれば消します．

1.2 CLEAR A ………例2

PB－300の各プログラムエリア（P 0～P 9）に，プログラムが記憶されていれば，すべてのプログラムを消し，プログラムエリアをP 0に指定します．

以上のコマンドは，プログラムを新しく作り始めるとき，あるいはプログラムが不要になったときなどに実行します．

ただし，このコマンドを実行した後，「しまった！」と思っても，プログラムは元に戻りません．十分注意してください．

メモ CLEARコマンドに類似したコマンドに，VACコマンドがあるが，VACは，変数エリアをすべて初期値（0，ヌル）に設定するコマンドである．

2 LIST
プログラムの表示

命令	LIST 〔行番号〕 LIST A	読み方：リスト ：リストエー
書式例	例1．LIST 例2．LIST 150 例3．LIST A	Aの語源：All 〔 〕内は省略してもよい
実行	MODE 0(RUN)，MODE 1(WRT)のいずれでも実行可能	

2.1 LIST ………例1

現在指定しているプログラムエリアに，プログラムが記憶されているなら，先頭の行番号からプログラムを表示します．

2.2 LIST 150 ………例2

LISTに続いて行番号を指定すると，現在のプログラムエリアの指定行番号(ここでは150)から，プログラムを表示します．

2.3 LIST A ………例3

PB－300に記憶されているすべてのプログラムを，P0からP9まで順に，それぞれ先頭の行番号から表示します．

2.4 RUNモードとWRTモードでの実行の違い

LISTコマンドは，RUNモード，WRTモードのいずれでも実行可能です．



RUNモードでの LIST または LIST Aコマンド，WRT モードのLIST Aコマンドを実行すると，プログラムが一定時間表示された後，連続して次々に表示されます．

WRT モードで，LISTコマンドを実行すると，1行ごとに表示して，カーソルが点滅します．

この状態でプログラムの修正が可能です．

プログラムを修正したら，EXEキーを押します．EXEキーを押すことにより，修正したプログラムがPB－300に記憶され，表示も次の行に変わります．修正の必要がない場合は，単にEXEキーを押すことにより，次の行に表示が変わります．

2.5 LIST，LIST Aの実行を中断する

LIST，LIST Aコマンドの実行の中断は，STOPキー，またはACキーを押すことで可能です．

STOPキーを押すと，表示はそのままで，自動的に行が変わる機能が中断するため，EXEキーを押すことにより再開します．

ACキーを押すと，表示が消え，LISTまたはLIST Aの実行状態から抜け出ます．

なお，WRT モードでプログラムの修正に関係ないキーの入力は，必ず，このACキーを押してからにしてください．

2.6 ミニプリンタへプログラムリストを出力する

LISTまたはLIST Aコマンドを実行する前に，MODE7を指定すれば，ミニプリンタへプログラムのリストが印字されます．

RUNモードでは，次々に印字していきますが，WRTモードでは，EXEキーを押すごとに，1行ずつ印字されます．

印字の様式も，RUNモードとWRTモードでは，次図のように異なります．

■RUNモードのリスト

```
LIST
  10 PRINT "TASIZAN"
  20 PRINT "A+B+C"
  30 INPUT "A=",A,"B
  =",B,"C=",C
  40 D=A+B+C
  50 PRINT "A+B+C=";
  D
  60 END

READY P0
```

■WRTモードのリスト

```
LIST
10 PRINT "TASIZAN"
20 PRINT "A+B+C"
30 INPUT "A=",A,"B="
,B,"C=",C
40 D=A+B+C
50 PRINT "A+B+C=";D
60 END

P _1___56789
```

3 RUN

プログラムの実行開始

命令	RUN 〔行番号〕	読み方：ラン
書式例	例1. RUN 例2. RUN 100	〔 〕内は省略してもよい
実行	MODE 0（RUNモード）のみ 実行可能	

3.1 RUN ………例1

現在指定されているプログラムエリアのプログラムで，最も行番号の小さい行から命令を実行します．

RUNを実行すると，それ以後の制御はプログラムで行ないます．

3.2 RUN 100 ………例2

RUNに続いて行番号を指定します．行番号は整数のみです．

このコマンドによって，現在指定されているプログラムエリアの指定行番号から実行します．この例では，行番号100から実行を始めます．

もし，100行がなければ，100行より大きく，100行に最も近い行番号から実行を開始します．100行より大きい行番号がなければ，何も実行せずにコマンド待ちに戻ります．

```
RUN ⇨             10  PRINT "A+B"                       10  PRINT "A+B"
(10行から実行)     20                                    20
                   :                                     :
RUN100 ⇨          100  INPUT A                          90  INPUT A
(100行から実行)    200  INPUT B          RUN100 ⇨        200  INPUT B
                   :                  (200行から実行)     :
                  500  END                              500  END
                                      RUN600 ⇨
                                      (何も実行しない)
```

3.3 RUNと同機能の操作 ………S 0 ～ 9

RUNモードにおいて，S（SHIFT）キーと0～9のうち1つのキーを押すと，指定されたプログラムエリアの最初の行から実行します．

たとえば，S 0なら，プログラムエリア0（P0）のプログラムの最初の行から実行します．この操作では，途中の行からの実行はできません．

4 DEFM
変数エリアの大きさを指定

命令	DEFM 〔拡張数〕		
書式例	例1. DEFM 例2. DEFM 20 例3. DEFM 0	実行後の表示	***VAR:50 ***VAR:46 ***VAR:26
実行	MODE 0 (RUNモード), MODE 1 (WRT モード) のいずれでも実行可能		

読み方：ディファイ ンメモリー

語源：DEFine Memory

VARの語源：VARiable

4.1 DEFM ………例1

変数エリアが現在いくつであるか，その大きさを表示します.

変数エリアの大きさの基本は26ですが，〔例1〕では，拡張されていたので，50と表示されたことになります.

4.2 DEFM 20 ………例2

DEFMに続いて，拡張する変数の大きさを入力します.

拡張できる変数の大きさは，メモリーの許す限りです. プログラムで多くのデータを扱う場合には拡張が必要ですが，プログラムのサイズが大きくなってくると，予定数の拡張ができなくなることがあります.

このような場合，データ数は減らさずに，プログラムの方をブロック単位に分割して，1つのブロックが終わったら，次のブロックをプログラムテープから読み出して処理する，ということを繰り返してプログラムの案を練り，その段階で予定していた結果を得るように工夫してください.

なお，変数エリアの拡張とステップ数との関係は，〔第1編〕 25ページにくわしく解説してあります.



〔例2〕のDEFM20を実行すると，基本26と拡張20の変数エリアを合わせて46が表示されます.

4.3 DEFM 0

前項の使い方の1つで，拡張された変数エリアを，基本の26へ戻すときに使います.

■1行は62文字まで■

PB−300では，計算式，プログラムともに，1行は62文字まで入力できます.

56文字以上になると，"■"のカーソルが点滅し始めて，「残りが少なくなった」ことを注意してくれます. そのときは，区切りのよいところでEXEキーを押して，途中までの計算結果を求め，または，1文を終了し，新たに計算式，プログラムを入力してください.

注1 ステップ数にはスペースは数えませんが，文字列中のスペースは数に入ります.

注2 プログラムの場合，1行62文字ギリギリでEXEキーを押した後に，その行のLISTを見ると，行番号とコマンド間のスペースが消えています. これは，文字を62文字まで入力すると，スペースが前の方から自動的に消えてつめられるようになっているからです.

次の1行を入力して，確かめてください.

```
10 PRINT␣"*CASIO␣PERSONAL␣COMPUTER␣
PB−300␣&␣CASSETTE␣INTERFACE"
```

以上を入力し，EXEキーを押すと……

```
10␣PRINT␣"*C
```

………と表示されます.

ところが，LISTをとると，……

```
10PRINT␣"*CA
```

……のように表示され，10とPRINTの間のスペースが消えています.

5 プログラムコマンド(1) 入力命令

1 INPUT

プログラムの実行は入力が終わるまで中断

命令	INPUT〔"文字列",〕変数名〔,〔"文字列",〕変数名〕
書式例	例1. INPUT A 例2. INPUT A, B 例3. INPUT "KAZU", C 例4. INPUT "1&2", D, E 例5. INPUT "1=", D, "2=", E
機能	キーボードからのデータを変数に入力する

読み方：インプット

〔 〕内は省略してもよい.

1.1 INPUT ………例1

実行時の表示 ⇨

```
?
```

入力命令の代表格がINPUTです.

INPUTは，命令を実行すると，入力要求のメッセージを必ず表示します. データは，キーボードを押して入力します.

メッセージを表示した後は，オペレーターがキーボードからデータを入力するのを，ずっと待っています. 忠実なPB−300は，浮気したり，よそ見したりはしません.

データの最後（区切り）は，EXEキーです. 数字，文字，記号のキーを押し，EXEキーを押して初めて，1つのデータがパソコンへ入力されたことになります. データは，INPUTに続いて書かれた変数に代入されます.〔例1〕では変数Aに代入されることになります.

では次に，〔例1〕のようなINPUT命令の使い方の注意を，事例をあげて述べておきます.

■ 使える変数は，数値変数，文字変数，専用文字変数，配列変数です.

文字変数 ………… INPUT B$
専用文字変数 …… INPUT $
配列変数………… INPUT L(4)

以上の，いずれの実行後の表示とも，次のようになります.

```
?
```

■ INPUT文で入力できるデータは，変数の扱える範囲です.

数値変数 ………… $\pm 1\times10^{-99}\sim\pm 9.99999999999\times10^{99}$ と 0
数値の演算式
A～Zの変数名1字（ただし，数値が記憶されていること）
配列変数

文字変数 ………… 7文字までの文字（数字も含む），記号
専用文字変数 …… 30文字までの文字（数字も含む），記号

次に，その事例を示します.

書式例	入力するデータ	結果
INPUT A	920 EXE 7.32E36 EXE 34+56+22 EXE	変数Aには920が入力される 〃 7.32×10^{36}が入力される 〃 112が入力される
INPUT B$	DATA EXE ␣␣␣␣␣␣␣ EXE	B$にはDATAが入力される 〃 スペース7個が入力される

■ 規格外のデータを入力した場合のエラーメッセージと対策法

書 式 例	入力するデータ	エラーメッセージ
INPUT A	1E-100 (1×10^{-100})	ERR 2
INPUT B$	ABCDEFGH（8文字）	ERR 6
INPUT $	AB…Z1…5（31文字）	ERR 6

＊エラー表示後の対策

AC キーを押すとエラーメッセージは消え，？を表示します．
この状態で，変数の許容範囲内のデータを入力してください．

■ 数値変数へ文字を，文字変数へ数値を入力するとどうなるか．

書 式 例	入力するデータ	結 果
INPUT A	D（変数Dの内容が160のとき）	変数Aに 160 が入力される
	DE	ERR 2
	E（文字変数E$の内容が"XYZ"のとき）	ERR 6
INPUT B$	120	文字列"120"がB$に入力される
	D（変数Dの内容が160のとき）	文字列"D"がB$に入力される
	E（文字変数E$の内容が"XYZ"のとき）	文字列"E"がB$に入力される

☆ ☆ ☆

プログラム 1 入力した数の倍々を計算

このプログラムは，ある数を入力すれば，EXE キーを押すごとに，前の表示の２倍の数を表示します．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

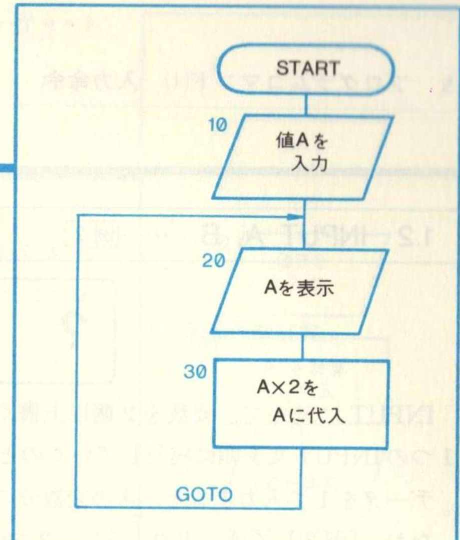
プログラムの実行

MODE 0
RUN EXE

実行時の表示

実行時の表示	キー入力
?	⇨ 1 EXE
1	⇨ EXE
2	⇨ 以下同じ

■フローチャート



プログラムリスト

```
10 INPUT A
20 PRINT A
30 A=A*2
40 GOTO 20
```

プログラム解説

10：入力命令．データは変数Aに入力する．

20：出力命令．変数Aを表示する．

30：演算．Aの値を２倍した後，再び変数Aに代入する．

40：分岐命令．プログラムの実行を20行へ移す．

このプログラムは，入力命令の他に，出力命令と分岐命令を使っています．計算はかけ算です．

■実行例

```
RUN

?
1
 1
 2
 4
 8
 16
 32
 64
 128
 256
 512
 1024
 2048
 4096
 8192
 16384
 32768
 65536
 131072
 262144
 524288
```

1.2　INPUT　A, B　……例2

実行時の表示 ⇨

?

INPUTに続いて，変数を2個以上書くことができます．その働きは，1つ1つのINPUT文を順に実行していくのと同じことです．

データを1つ入力すると，次の変数分の入力要求“?”が表示されます．

なお，〔例2〕では，次のように，2つのINPUT文が1つにまとめられたと考えることができます．

```
INPUT A
INPUT B   } ⇨ INPUT A, B
```

では，この書式の使い方の注意を，次に述べます．

■ 変数の書き方

① 変数と変数の区切りはカンマ（,）である．

② 変数はどのような組み合わせ，順序でも書くことができる．

```
INPUT Z$, C, $, F
```

実行すると

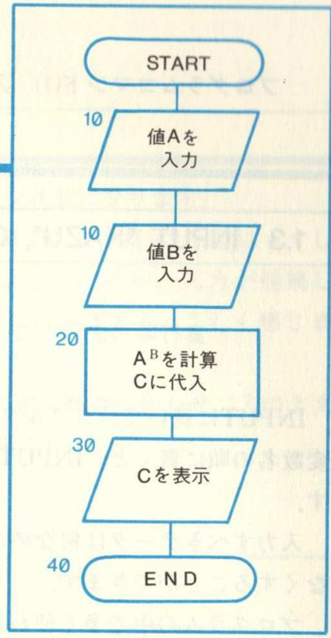
⇩

表　示	キー入力	結　果
?	END EXE	変数Z$にENDを入力
?	366 EXE	変数Cに366を入力
?	SAMPLE EXE	変数$にSAMPLEを入力
?	−9420 EXE	変数Fに−9420を入力

■ データの入力範囲とエラー

前項のINPUT Aの書式と同じです．

■フローチャート



プログラム　2　x^yの計算

このプログラムは，2つの数を入力すると，先に入力した数を，後で入力した数でべき乗して表示します．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

実行時の表示	キー入力
?	⇨ 2.7 EXE
?	⇨ 5 EXE
143.48907	

プログラムリスト

```
10  INPUT A, B
20  C=A↑B
30  PRINT C
40  END
```

プログラム解説

10：入力命令．変数Aと変数Bにデータを入力する．

20：演算．AのB乗を計算してCに代入する．

30：出力命令．Cの値を表示する．

40：END文．プログラムの終了

このプログラムは，入力→演算→出力のパターンです．べき乗は“↑”を用います．

1.3 INPUT "KAZU", C ………例 3

実行時の表示 ⇨

```
KAZU ?
```

INPUTに続いて，ダブルクォーテーション(")で囲んだ文字列，カンマ(,)，変数名の順に書くと，INPUT 文実行時には，文字列と入力要求？を表示します．

入力すべきデータは何なのかを表示しますので，入力するデータの誤りを少なくすることができます．

プログラムの中で多く使われるINPUT 文の書式は，この形です．もう一度この書式を示すと，次のようになります．

最も一般的なINPUT文の書式　INPUT "DATA" , Z (I)
（"DATA"…"文字列"，","…，，Z (I)…変数名）

次に，この書式を使うときの注意を述べます．

■ 文字列の文字数

① BASICの1行は，62文字まで書くことができます．62文字の中には，行番号，コマンド，変数，記号も含みます．

また，文字列の文字数は，30文字までです．

② 実用上の文字数――11文字まで

次の2行で，表示の仕方，時間などを比較してください．

```
100 INPUT "TENSU␣INPUT", T
200 INPUT "TENSUU␣NO␣INPUT:", T
```

100行

```
TENSU  INPUT ?
```

200行

スクロール

```
TENSUU  NO  INPUT : ?
```


100行は瞬時に表示して，入力が可能な状態になります．

それに比べ，200行は，TENSUU NO IN まで表示し，しばらくしてからPUT：？ を表示します．このように，表示，入力が連続しないので，多くのデータを入力する際には，まどろっこしく感じます．

したがって，実用上の文字数は，表示部の桁数に合わせ，？の1文字分を引いた11文字以内にした方が便利です．

■ 使用できる変数，データの入力範囲，エラー

〔1.1〕項の INPUT A と同じです．

☆　　　☆

プログラム 3　加減算

このプログラムは，数値データを入力すると，それまで入力したデータの総和を表示します．

＋，－のいずれの数値でも入力可能です．また，データの入力時に，何件目のデータであるかも表示します．

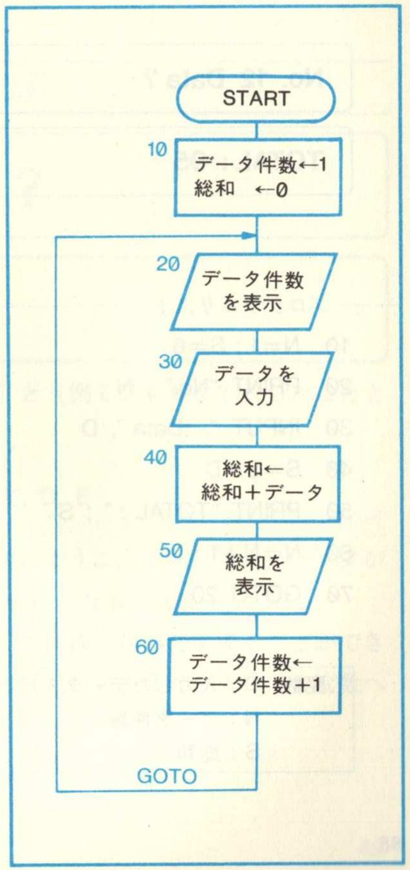
プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

■フローチャート



実行時の表示	キー入力
No. 1 Data ?	⇨ 24 EXE
TOTAL : 24	⇨ EXE
No.2 Data ?	⇨ 54 EXE
TOTAL : 78	⇨ EXE
⋮	
No. 12 Data ?	⇨ 47 EXE
TOTAL : 95	⇨ EXE
⋮	⋮

プログラムリスト

```
10  N=1 : S=0
20  PRINT "No." ; N ;
30  INPUT "␣Data ", D
40  S=S+D
50  PRINT "TOTAL :" ; S
60  N=N+1
70  GOTO 20
```

変数	
D	入力したデータ
N	データ件数
S	総和

プログラム解説

10 : 初期設定．Nはデータ件数，Sはデータ合計

20 : 出力命令．データのNo.を表示．INPUT文を続けて表示するため，最後は；

30 : 入力命令．文字列は“␣Data ”です．データはDに入力される．

40 : 演算．データの総和を計算する．

50 : 出力命令．計算した総和Sを表示する．



60 : 演算．データの件数に1を加える．

70 : 分岐命令．プログラムの実行を20行へ移す．

このプログラムは，INPUT文とPRINT文を組み合わせて，入力時のメッセージを表示させています．PRINT文についての詳細は出力命令の節で解説します．

1.4　INPUT “1&2”, D, E　………例 4

実行時の表示 ⇨

1&2 ?
?

この書式例は，次に示すように，〔例3〕と〔例1〕を順に実行したことと同じです．

```
INPUT "1&2", D
INPUT E          } ⇨ INPUT "1&2", D, E
```

このINPUT文を実行すると，上記の表示のように，1回目はメッセージが表示されますが，2回目以後は入力要求 ? のみが表示されます．

この例では，変数が2個ですが，〔例2〕と同様に何個でも書くことができます．その際，データは前から順に書かれた変数に入力され，すべての変数へデータが入力されないとINPUT文は終了しません．

では次に，この書式の使い方の注意を示します．

■ 文字列の書き方

〔1.3〕項の INPUT "KAZU", C と同じです。

なお，変数の1個目への入力時だけ文字列（メッセージ）が表示されます。

■ 変数の書き方

① 文字列と変数，変数と変数の区切りはカンマ（,）です。

② 変数はどのような組み合わせ，順序でも書くことができます。

INPUT "DATA :", D$, E, $, A

実行すると ⇩

表　　示	キー入力	結　　果
DATA : ?	INPUT EXE	変数D$にINPUTを入力
?	6 E 10 EXE	変数Eに6×10^{10}を入力
?	NAMAE wa EXE	変数$にNAMAE waを入力
?	7.2 EXE	変数Aに7.2を入力

■ データの入力範囲，エラー

〔1.1〕項の INPUT A と同じです。

☆　☆　☆

プログラム 4　比較計算プログラム

このプログラムは，5件のデータを入力して，その合計と構成比を表示します。

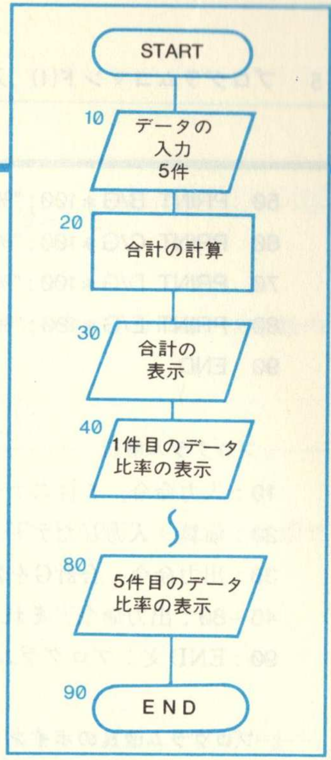
プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

■フローチャート



実行の表示	キー入力
Data 1 to 5 : ?	⇨ 76 EXE
?	⇨ 33 EXE
?	⇨ 61 EXE
?	⇨ 89 EXE
?	⇨ 17 EXE
Gokei 276	⇨ EXE
スクロール 27.53623188%	⇨ EXE
スクロール 11.95652174%	⇨ EXE
スクロール 22.10144928%	⇨ EXE
スクロール 32.24637681%	⇨ EXE
6.15942029%	⇨ EXE プログラム終了

プログラムリスト

```
10  INPUT "Data 1 to 5:", A, B, C, D, E
20  G=A+B+C+D+E
30  PRINT "Gokei"; G
40  PRINT A/G*100; "%"
```

```
50 PRINT B/G*100;"%"
60 PRINT C/G*100;"%"
70 PRINT D/G*100;"%"
80 PRINT E/G*100;"%"
90 END
```

変数A: 1件目のデータ
B: 2件目のデータ
C: 3件目のデータ
D: 4件目のデータ
E: 5件目のデータ
G: 合計

プログラム解説

10：入力命令．5件のデータを入力して，変数AからEに記憶する．

20：演算．入力したデータを合計して，Gに代入する．

30：出力命令．合計Gを表示する．

40〜80：出力命令．それぞれの変数と合計の比を%で計算し表示する．

90：END文：プログラムの終了．

プログラム改良のポイント

このプログラムは，データの記憶に使う変数を中心にして，入力方法，演算方法，出力方法を改良できます．

出力（表示）は，多数桁は必要ないので，SET F コマンド，または RND で必要な桁数の表示に変えてください．

1.5 INPUT "1=", D, "2=", E ………例 5

実行時の表示 ⇨

1=?

2=?

この書式は，次のように，〔例3〕を複数個1行に書いたものです．文字列と変数が1対1で書かれているので，データ入力の都度，メッセージと入力要

INPUT "1=", D
INPUT "2=", E
} ⇨ INPUT "1=", D, "2=", E

求の ? を表示します．

実行の順は，文字列と変数の1組ずつ前の方からです．1行には，文字数の制限内で何組でも書くことができます．

次に，この書式の使い方の注意を述べておきます．

■ 文字列と変数の書き方

① 文字列の文字数は，〔1.3〕項の INPUT "KAZU", C と同じ注意で書いてください．

② この書式をまとめると，次のようになります．

INPUT "1=", D , "2=", E

（1つの入力文 , 1つの入力文）

③ 文字列と変数を1組とする入力文の間はカンマ（,）で区切ります．

④ 変数は，1文に数値，文字変数が混在でき，順番の制限もありません．

INPUT "DATA1:", Z$, "SUU:", L, "DATA2:", $, "SUU:", M

実行すると ⇩

表示	キー入力	結果
DATA1:?	TOKYO EXE	変数Z$にTOKYOを入力
SUU:?	7550 EXE	変数Lに7550を入力
DATA2:?	OOSAKA EXE	変数$にOOSAKAを入力
SUU:?	6900 EXE	変数Mに6900を入力

■ データの入力範囲，エラー

〔1.1〕項の INPUT A と同じです．

プログラム 5 文字の組み合わせ

このプログラムは，4種類の文字を入力して，さまざまな文字の組み合わせを表示します．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行の表示

表示	入力
7 moji made	⇨ EXE
1 wa ?	⇨ GAME EXE
2 wa ?	⇨ PASOCON EXE
3 wa ?	⇨ RADIO EXE
4 wa ?	⇨ PENCIL EXE
RADIO GAME	⇨ EXE
PASOCON GAME	⇨ EXE
GAME PENCIL	⇨ EXE

プログラムリスト

```
10  PRINT "7 moji made"
20  INPUT "1 wa", T$(1), "2 wa", T$(2), "3 wa", T$(3),
    "4 wa", T$(4)
30  A=INT (RAN#*4)+1
40  B=INT (RAN#*4)+1
50  PRINT T$(A); "␣"; T$(B)
60  GOTO 30
```

変数A：整数1から4．配列の引数
B：整数1から4．配列の引数
T$(1)～T$(4)：入力したデータ

プログラムの解説

10：出力命令．7文字までの入力であることを表示する．

20：入力命令．配列 T$(1)から T$(4)へ文字列を入力する．

30, 40：演算．整数1～4を乱数から作って，AとBに代入する．

50：出力命令．配列のA番目とB番目の内容を並べて表示する．

60：分岐命令．プログラムの実行を30行へ移す．

このプログラムは，文字の制限を表示した後，文字列4個を入力します．入力した文字は配列に代入されます．次に，乱数により，入力したうちの2つの文字列を続けて表示させます．

プログラム改良のポイント

多くの文字を入力して，その中から文字を選択して組み合わすと，楽しい言葉が生まれます．

入力方法と配列の方法がポイントです．

2 KEY

入力チャンスは1回限り

命令	KEY	読み方：キー
書式例	Z$=KEY	

機能 命令が実行されているとき，押されたキー1文字を変数へ代入する

2.1 データ入力

KEYコマンドでデータを入力できるのは，KEYコマンドを実行しているときだけです．押したキー（データ）は文字として扱われるので，書式例のように，文字変数＝KEYの代入式の形で使います．

入力できるデータは，1文字に限られています．

2.2 データ未入力

KEYコマンドを実行しているとき，キーが押されていなければ，文字変数には何も入力されない（ヌル）状態で，次に書かれている命令の実行へ移ります．

2.3 命令実行時の表示

KEYコマンドを実行しても，表示には何も現われません．また，KEYコマンドを実行する前に表示されていても，表示は変化しません．

2.4 INPUT命令とKEY命令の違い

	KEYコマンド	INPUTコマンド
命令実行時の表示	何も表示しない	?（入力要求メッセージ）
データの入力	命令実行時に押されたキー 押されていなければ未入力のまま	EXEキーが押されるまでに入力したデータ
データの種類	文字変数，1文字	数値，文字変数，変数の範囲内
次の命令の実行	そのまま即時	データ入力終了まで中断

メモ KEYコマンドを使用したプログラム例は，212ページにまとめて示しておきました．

〔練習問題1〕 合計と平均を求めるプログラム．

数を入力したら，これまで入力した数の合計と平均を表示するプログラムを作ってください．

平均の表示は小数点第1位までとしてください．

合計と平均を表示したら，再び数の入力へ戻り，繰り返してください．

答えは226ページ⇨

6 プログラムコマンド(2) 出力命令

1 PRINT
データを表示部へ出力

命令		読み方
PRINT	出力命令	: プリント
CSR n	表示位置指定	: カーソル
SET En	表示桁数指定（有効桁数）	: セット イー
SET Fn	表示桁数指定（小数点以下）	: セット エフ
SET N	表示桁数指定解除	: セット エヌ

1.1 PRINTとCSRはペア，SETは独立

PRINTは，表示部へ定数や変数を表示するコマンドで，多様な使い方があります．

CSRは，PRINTコマンドとペアで書き，CSRの次のnの値で表示位置を決めます．

SETには3種類の書き方があり，数値を表示するときの桁数を指定します．

SET Enは，有効桁数を指定します．SET Fnは，小数点以下の桁数を指定します．SET Nは，SET EとSET Fの指定を解除します．

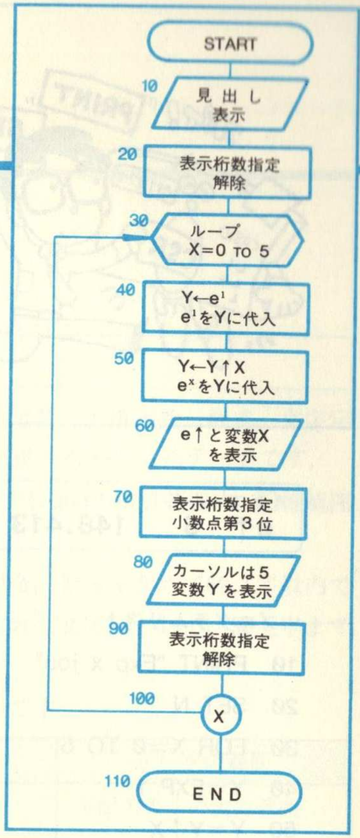
なお，SETコマンドは，プログラム中の必要な行に独立した命令として書きます．

1.2 PRINT, CSR, SETのプログラム中での書き方

PRINT命令はCSR，SETの各命令と併用することで，便利な命令に成長します．

PRINT	CSR	SET
		80 SET F3
100 PRINT A	100 PRINT CSR 0;A	100 PRINT A
		150 SET N

■フローチャート



プログラム 6 自然対数の底eの0～5乗の計算

このプログラムは，RPINT, CSR, SETの各コマンドを使った例です．

小数点以下の表示桁数は，プログラム中で3桁に指定します．

べき乗した値を表示する位置もCSRでそろえます．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

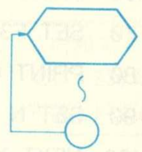
プログラムの実行

MODE 0
RUN EXE

実行の表示

表示	キー入力
Exp x jou	EXE
e↑ 0 1.000	EXE
e↑ 1 2.718	EXE
e↑ 2 7.389	EXE
e↑ 3 20.086	EXE
e↑ 4 54.598	EXE

注 ループ命令の記号



…は，正しい記号ではありません．ループ命令をわかりやすく表現するために，便宜上使っています．



スクロール
e↑　5　148.413

⇨ EXE　プログラム終了

プログラムリスト

```
10  PRINT "Exp x jou"
20  SET N
30  FOR X=0 TO 5
40  Y=EXP 1
50  Y=Y↑X
60  PRINT "e↑"; X;
70  SET F3
80  PRINT CSR 5; Y
90  SET N
100 NEXT X
110 END
```

変数X：べき乗数
　　Y：e^x の計算結果

プログラム解説

10：出力命令．文字列の表示．
20：表示桁数指定解除．
30：ループ命令．Xには0から5まで，1つずつ代入される．
40：演算．e^1 を求めて，変数Yに代入する．
50：演算．eのX乗を変数Yに代入する．
60：出力命令．文字列e↑と変数Xを表示する．この後に続けて表示するデータがあるので最後は；
70：表示桁数を小数第3位に指定する．
80：出力命令とCSR命令．表示は6番目から変数Yを表示する．
90：表示桁数指定解除．
100：ループ命令折り返し
110：END文．プログラム終了

2 PRINT

PRINT命令の使い方

2.1 PRINT命令のルール

PRINT命令で表示部へ表示するのは，数値定数，数値変数，数式，文字定数，文字変数，文字式が，単独でまたは複数個組み合わさったデータです．

数値定数や変数，数式の表示できる範囲は，PB-300の演算範囲内，記憶範囲内に限定されます．

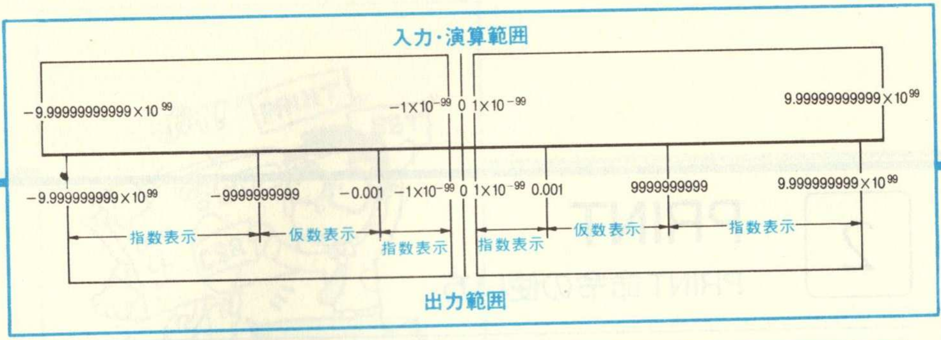
文字定数は，プログラムの1行が行番号，命令，記号を含めて62文字以内であれば，ダブルクォーテーション（”）で囲まれた文字列として，30文字まで書けます．

次の表は，これらの使い方例をまとめたものです．

データの種類	プログラムの書式	データの範囲
数値定数	PRINT 14.6E32	下記の表1の出力範囲
数値変数	PRINT A	〃
数式	PRINT 37/24+A	〃
文字定数	PRINT "MOJIRETSU"	1行62文字 ” 内30文字
文字変数	PRINT N$	文字変数7文字，$30文字
文字式	PRINT N$+$+"NG"	文字定数，変数と同範囲

■表1　プログラム計算の入力演算，出力範囲

	仮数部	指数部	範囲
入力・演算	12桁	2桁	0と$\pm1\times10^{-99}\sim\pm9.99999999999\times10^{99}$
出力	10桁	——	0と$\pm0.001\sim\pm9999999999$
出力	10桁	2桁	0と$\pm1\times10^{-99}\sim\pm9.999999999\times10^{-4}$ $\pm1\times10^{10}\sim\pm9.999999999\times10^{99}$



2.2 区切り符号；と，

PRINT命令で，いくつかのデータを続けて書く場合には，次のように，セミコロン（；）かカンマ（，）で，データどうしを区切ります．

PRINT　データ ｛；または，｝ データ………

セミコロン（；）は，前のデータに続けて後に続くデータを表示します．この場合，後に続くデータは，次の例のように，同じ行に書かれていなくてもかまいません．

```
100 PRINT "DATA"; A   ⇔   100 PRINT "DATA";
                          110 PRINT A
```

いずれの書き方でも，表示結果は，次のようになります．

表示結果　| DATA　300 |　……変数Aにデータ300が入力されている場合

次に，カンマ（，）は，前のデータを表示したら，プログラムが中断します．EXEキーを押すことで，後に続くデータを表示します．

```
100 PRINT "DATA", A   ⇔   100 PRINT "DATA"
                          110 PRINT A
```

上記の2つの書き方は，同じ表示結果となります．

表示結果　| DATA |　⇨ EXE

| 300 |

データの後に，；も，もなければ，データを表示してプログラムは中断します．次に書かれている命令を実行するには，EXEキーを押してプログラムを再

開します．

プログラムが中断しているときは，表示部の右上に小さくSTOPが表示されています．

```
100 PRINT "DATA"
```

実行時の表示　| DATA　STOP |　……EXEキーを押すと再開され STOPは消える

3 CSR

表示位置の指定

PRINT命令で，データを表示する位置を指定するときにCSRを使います．

PRINT CSR n ｛；または，｝

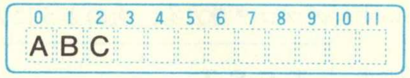
表示部には，左から，0，1，……，11の番号が付けられています．CSR nでこの番号を指定すると，その番号の位置からデータを表示します．



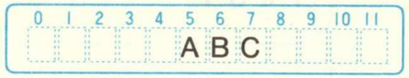
文字の表示は，指定した位置へデータの先頭がくる

正の数を表示すると指定した位置は，符号のために1字分スペースがとられる

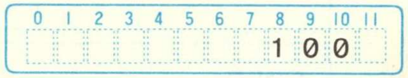
PRINT CSR 0；"ABC"



PRINT CSR 5；"ABC"



PRINT CSR 7；100



表示する位置は，0から11の間で，数値定数，数値変数，数式などで指定します．0より小さい数か，11より大きい数を指定するとエラー（ERR 2）になります．次の表に，指定の例を示します．

	書　式　例	数値の指定方法
数値定数	CSR 4	小数点以下は切り捨て
数値変数	CSR N	0.1→0, 11.9→11
数　　式	CSR L/2	n<0, n>11 はERR 2

次に，CSRの後のセミコロン（；）は，後に書かれたデータを指定した位置から表示します．後に書かれるデータは，CSRと同じ行の中でなくてもかまいません．

```
100 PRINT CSR 4;"TENSUU"  ⇔  100 PRINT CSR 4;
                             110 PRINT "TENSUU"
```

上記のいずれの書き方でも，表示結果は同じです．

表示結果　（0 1 2 3 4 5 6 7 8 9 10 11）　TENSUU

また，カンマ（，）は，前の表示を消して，プログラムを一時中断します．EXE キーを押すことでプログラムを再開しますが，データは，CSR nでどんな位置に指定してあっても，左端からの表示となります．

```
 90 PRINT "TEST"           ⇔   90 PRINT "TEST"
100 PRINT CSR 5,"END"          100 PRINT " "
                               110 PRINT "END"
```

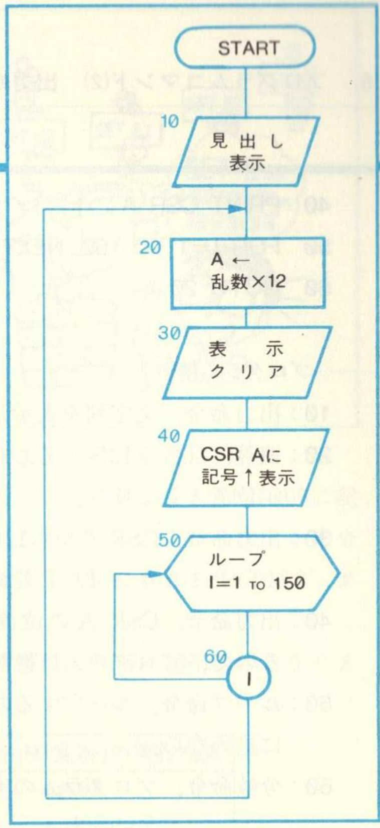
上記のいずれのプログラムも，次のように表示されます．

表示結果　TEST　⇨ EXE

（空白）　⇨ EXE

END

■フローチャート



プログラム 7 記号をランダムな位置に表示する

このプログラムは，PRINT命令とCSRの組み合わせで，記号をランダムな位置に表示し続けます．

表示する位置は乱数で決め，0～12未満の乱数を小数点が付いたまま使います．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

＊CSR SAMPLE＊　⇨ EXE

↑

↑

↑

以下自動的に連続して↑印を表示し続ける
プログラムを中断するときは STOP キーを押す

プログラムリスト

```
10 PRINT "*CSR SAMPLE*"
20 A=RAN#*12
30 PRINT CSR 0;"␣␣␣␣␣␣␣␣␣␣␣␣";
```

```
40 PRINT CSR A;"↑";
50 FOR I=1 TO 150:NEXT I
60 GOTO 20
```

プログラム解説

10：出力命令．文字列を表示して実行を中断する．

20：演算．乱数を12倍して，0以上，12未満の数をAに代入する．Aは表示位置として使う．

30：出力命令．CSR 0から12個のスペースを表示して，表示部をクリアする．セミコロン（；）で終っているので，プログラムは継続する．

40：出力命令．CSR Aの位置に，記号↑を表示する．（；）で終っているので，プログラムは継続する．

50：ループ命令．ループするだけで，一定時間だけプログラムの実行を先に進めない．

60：分岐命令．プログラムの実行を20行へ移す．

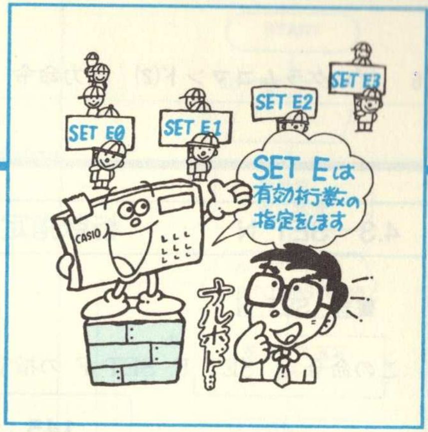
以上のように，記号を表示する位置は乱数で決めます．表示のクリアはスペース12個で行ない，表示をクリアした後，一定の時間，記号を表示し，再び乱数の発生から繰り返します．

表示する時間は，ループ命令の終値の設定で決まります．

4 SET
数値データの桁数指定

数値データを表示するときの桁数を指定する命令で，3つの形があります．

SET E ……… 有効桁数の指定
SET F ……… 小数点以下の指定
SET N ……… 桁数指定の解除



4.1 SET E ……… 有効桁数の指定

書式 SET E {0~9}

この命令により，有効桁数を指定すると，表示する値はすべて指数表示に変わります．

表示する値は，指定した桁＋1桁目で四捨五入されていますが，PB-300に記憶されている値は変わりません．あくまでも，表示する桁数だけを変える命令です．桁数は，1桁（SET E1）から，9桁（SET E9），10桁（SET E0）までを数値定数で指定します．

なお，プログラム計算では，SET E9，SET E0 で9桁，10桁表示となりますが，マニュアル計算では8桁表示になります．

桁数の指定	変数の値	(PRINT変数)の実行結果
SET E1	145	1E 02
SET E4	145	1.450E 02

4.2 SET F ……… 小数点以下の指定

書式 SET F {0~9}

小数点以下何桁まで表示するかを指定します．

表示する値は，指定した桁＋1桁目で四捨五入されますが，PB-300に記憶されている値は変わりません．

桁数は，整数部のみ（SET F0）から9桁（SET F9）までを，数値定数で指定します．

桁数の指定	変 数 の 値	(PRINT変数)の実行結果
SET F0	6.6666666666	7
SET F3	145	145.000
SET F9	6.6666666666	6.666666667

4.3 SET N ……… 桁数指定の解除

書式 SET N

この命令で, SET E, SET F の指定を解除します.

145 ⇨ **145**

6.6666666666 ⇨ **6.666666667**

4.4 SET E を指定し SET F を指定する

SET E が指定されている状態で, SET F を指定しても, エラーは発生しません. SET F が指定された後は, 小数点以下の桁数指定だけが有効となり, SET E は解除されます.

SET F が指定されている状態で, SET E を指定しても同様です.

☆ ☆ ☆

プログラム 8 sin0° から180° の計算

このプログラムは, sinの計算をします. 計算した値は収束しないものもあるので, 小数第3位までの表示としています.

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行後の表示

Sin ⇨ EXE 以下同じ

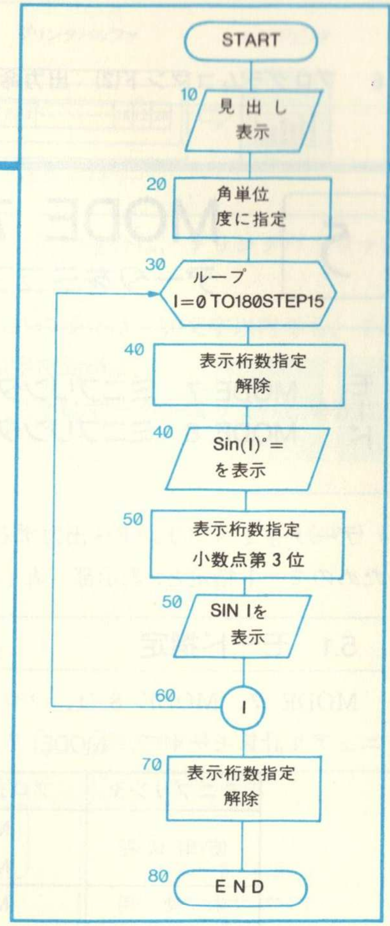
スクロール
sin 0°= 0.000

スクロール
sin 15°= 0.259

⋮

スクロール
sin 180°= 0.000

■フローチャート



プログラムリスト

```
10 PRINT "Sin"
20 MODE 4
30 FOR I=0 TO 180  STEP 15
40 SET N:PRINT "Sin";I;
   "°=";
50 SET F3:PRINT
   SIN I
60 NEXT I
70 SET N
80 END
```

プログラム解説

10：出力命令. Sin を表示して実行を中断する. EXE キーでプログラム再開する.

20：角度単位指定. 度 (DEG)

30：ループ命令. 0～180を15おきに変数Iに代入して実行する.

40：出力命令. 桁数の指定を解除した後, PRINT命令を実行し, sin何度かを表示する. 最後は (;) であるのでプログラムは継続する.

50：出力命令. 表示桁数は小数点第3位に指定する. その後, 40行の表示に続いて, SIN I の計算結果を表示する. この後, プログラムは中断するので, EXE キーを押して再開する.

60：ループ命令折り返し

70：表示桁数指定解除

80：END命令. プログラムの終了

表示は, 角度は通常の表示, 計算値は小数点以下3桁となります.

5 MODE 7
データをミニプリンタへ出力

モード

MODE 7 ミニプリンタ使用状態（PRT表示）
MODE 8 ミニプリンタ未使用

データをミニプリンタへ出力するには，ミニプリンタのON, OFF切り換えのためのモード指定と，表示部へ表示するときに使う命令を組み合わせて用います．

5.1 モード指定

MODE 7，MODE 8 は，プログラム中に書いて指定するか，あるいはマニュアル計算モードで，MODE 7，MODE 8 を入力するかのいずれかです．

ミニプリンタ	プログラムで指定	マニュアル計算モードで指定
使用状態	MODE 7 MODE N	MODE 7
未使用 （使用終了）	MODE 8 MODE N	MODE 8

＊N は変数で，7 または 8 が代入されているとする

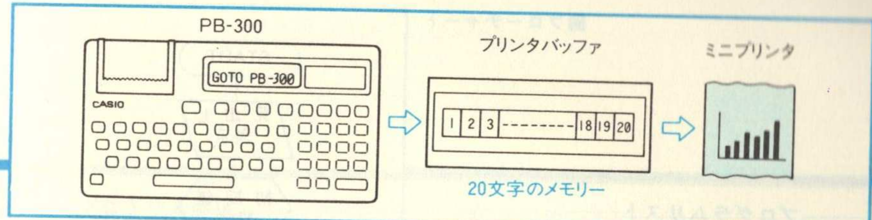
＊プログラム中のモード指定は，MODE キーではなく，M O D E とひとつずつキーを押してください．

5.2 データの印字

ミニプリンタの1行は，20文字まで印字できます．

データの印字は，次のページの図のように，データを一旦，20文字分のメモリーをもったプリンタバッファに入れてから行ないます．

したがって，MODE 7でPRINT命令を実行した場合，データの印字開始は，次の3つの形に分かれます．



① PRINT 命令の最後がセミコロン（；）の場合は，プリンタバッファに21文字目が送られたとき，印字を始める．

② PRINT 命令の最後がセミコロン以外の場合は，20文字以内でも，1単位のデータがバッファに入ったとき，印字を始める．

③ プログラム終了時に，プリンタバッファにデータが残っている場合は，終了と同時に印字を始める．

なお，プリンタバッファは，1回印字すると空（何も記憶されていない）になります．

☆ ☆ ☆

プログラム 9 SIN ウェーブを印字

このプログラムは，ミニプリンタへサインウェーブを印字します．

プログラム準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

Sin wave → キー入力 EXE

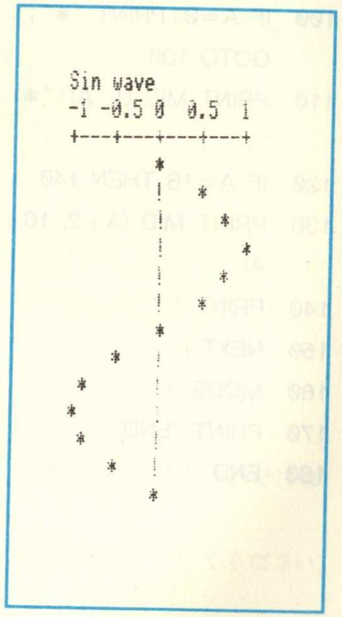
↓

ミニプリンタへ印字

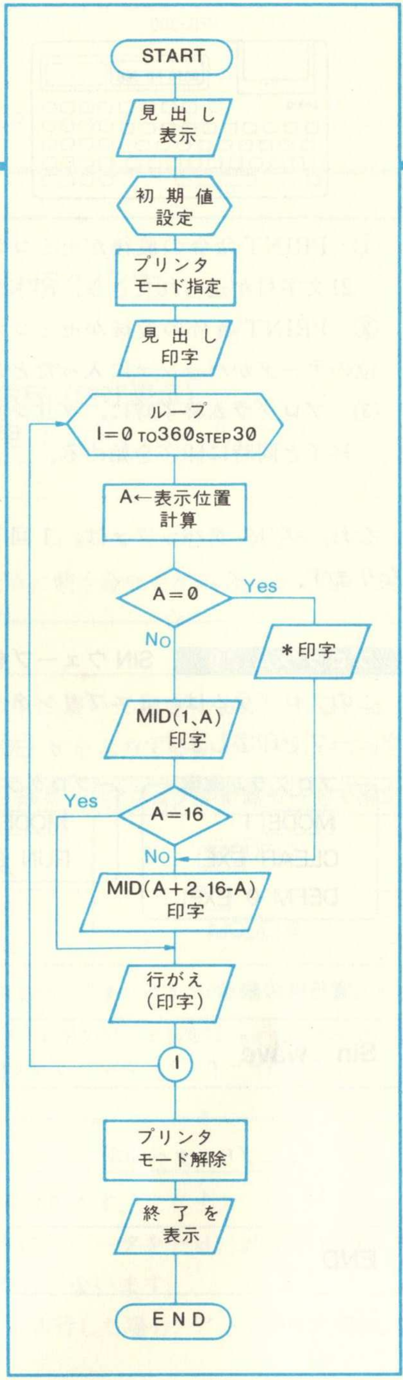
↓

END — プログラム終了

■印字結果



■フローチャート



プログラムリスト

```
10  PRINT "Sin wave"
20  MODE 4
30  $="␣␣␣␣␣␣␣␣!␣␣␣␣␣␣␣␣"
40  MODE 7
50  PRINT "Sin wave"
60  PRINT "−1␣␣−0.5␣␣0␣␣␣0.5␣␣␣1"
70  PRINT "+−−−+−−−+−−−+−−−+"
80  FOR I=0 TO 360 STEP 30
90  A=INT((SIN I+1)*8)
100 IF A=0;PRINT "*";:GOTO 130
110 PRINT MID (1, A);"*";
120 IF A=16 THEN 140
130 PRINT MID (A+2, 16−A);
140 PRINT " "
150 NEXT I
160 MODE 8
170 PRINT "END"
180 END
```


プログラム解説

10：出力命令．表示部に表示する．
20：角度単位指定．度（DEG）に指定する．
30：$に1行のフォーマットを代入する．
40：プリンタモード指定．プリンタ使用状態．
50：出力命令．プリンタへ印字する．
60, 70：出力命令．グラフのスケールと目盛を印字する．
80：ループ命令．値を変数Iに代入して実行する．
90：演算．SIN Iの印字位置を計算する．
100：条件処理．Aが0の場合は＊を印字して，130行へ分岐する．
110：出力命令．$の内容を1番目からA個と，＊印を印字する．
120：条件分岐．Aが16なら140行へ分岐する．
130：出力命令．＊印以降を印字する．
140：出力命令．1行を印字する．
150：ループ命令折り返し．
160：プリンタモード解除
170：出力命令．END を表示する．
180：END命令．プログラムの終了

変数	A：印字桁数，0～16の値
	I：角度，ループカウンタ
	$：1行の印字フォーマット
	"␣␣␣␣␣␣␣␣!␣␣␣␣␣␣␣␣"

1行の印字フォーマットを$に代入しておきます．

SIN Iの値をグラフを描くのに都合のいい値に変換して，$と組み合わせて印字します．

80行ループのステップ数と90行SINを他の値，関数に変えて試してください．

☞ 〔練習問題 2〕 円の円周と面積，球の体積を求めるプログラム

半径を入力して，円の円周と面積，球の体積をプリンタへ印字するプログラムを作ってください．

公　式　円の円周：2＊π＊R　($2\pi R$)

円の面積：π＊R↑2　(πR^2)

球の体積：4/3＊π＊R↑3　($4/3\pi R^3$)

ただし，半径をRとする

印字位置は次のとおりとします．

① 1桁目から7桁目に半径，円周などの文字列

② 8桁目に区切りの：印

③ 9桁目に計算結果を小数点第2位まで印字

印字例　HANKEI␣：　4．2

ENSHUU␣：　26．38

（␣ このマークはスペースを表わす）

答えは 226 ページ⇨

7 プログラムコマンド(3) 分岐命令

1 GOTO, IF～THEN
プログラムの流れを変える

前章までで，プログラムの2大ポイントである入力と出力に関する命令を解説してきました．

これからの命令は，プログラムのどこに使われるという決まりはなく，あらゆる場所で他の命令と関係し合って，プログラムを構成します．

プログラムは，いろいろな命令の集合なので,命令どうしをどう組み合わせるかが，プログラムをスッキリ作るポイントになります．

そういう意味では，GOTO 命令とIF～THEN 命令などの分岐命令の使い方は，プログラム作成上の分れ道になりかねません．特長をよくつかんで，うまく活用してください．

命令	
	GOTO（行番号またはプログラムエリア）……（無条件分岐）
	IF（条件式）THEN（行番号またはプログラムエリア）…（条件分岐）

読み方：ゴートゥー

：イフ～ゼン

1.1 GOTOとIF～THENの違い

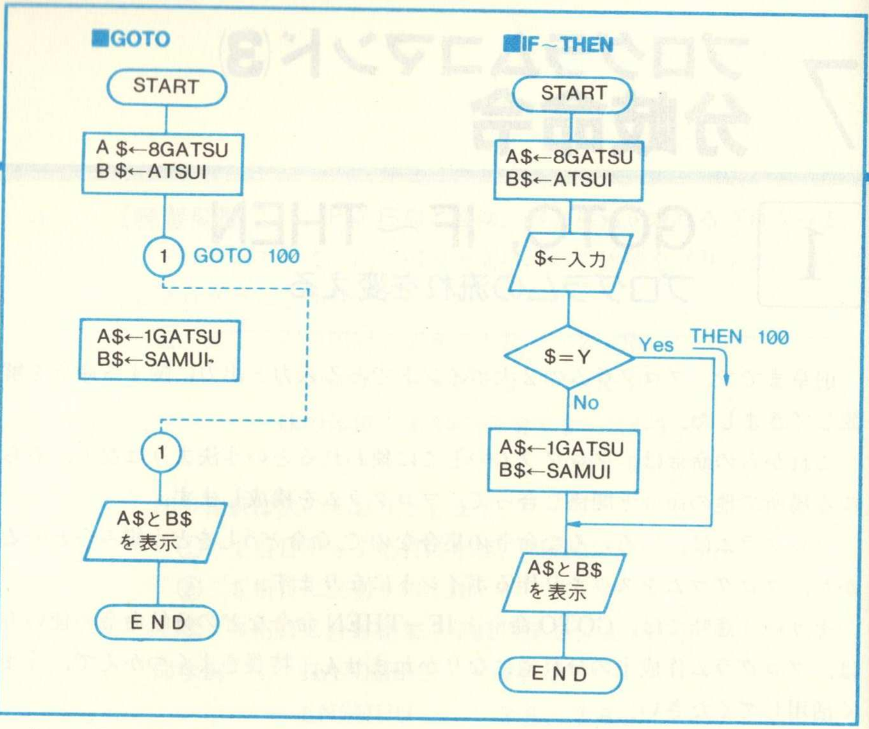
GOTO命令を実行すると，後に書かれた行番号またはプログラムエリアへ必ず分岐します．

たとえば，下のプログラムでは，20行を実行後100行を実行するので，表示部は，［8GATSUATSUI］が表示されます．

GOTOのプログラム⇨

```
10  A$="8GATSU":B$="ATSUI"
20  GOTO 100        ……無条件に100行へ飛ぶ
30  A$="1GATSU":B$="SAMUI"
100 PRINT A$;B$
110 END
```

■フローチャート



次に，IF~THENでは，条件式が成立するかどうかで，その後の実行が変わります．この点がGOTO命令とIF~THEN命令の大きな違いです．

下のプログラム例では，「$の内容がYである」ことが条件です．$にYが記憶されていたら，THEN以後（100へ分岐）を実行するし，$の内容がY以外なら，次の行（40行）を実行します．

IF~THENのプログラム ⇨

```
 10 A$="8GATSU":B$="ATSUI"
 20 INPUT " → Y/N ", $
 30 IF $="Y" THEN 100
 40 A$="1GATSU":B$="SAMUI"
100 PRINT A$;B$
110 END
```

GOTOとIF~THENの2つのプログラムをフローチャートで表わすと，上の図のように書けます．なお，GOTOのフローチャートは，わかりやすくするために，他のフローチャートの書き方と異なっています．

1.2 GOTOとIF~THENの分岐先

GOTO命令とIF~THEN命令は，希望するところへ分岐できますが，分岐の仕方が2通りあります．

(1) 行番号の分岐

同じプログラムエリア内なら，指定した行番号へ分岐して，プログラムを継続実行します．

GOTO 100 ……… 100行へ分岐

IF 条件式 THEN 400 ……… 400行へ分岐

(2) プログラムエリアの分岐

プログラムエリアの分岐は，指定したプログラムエリアの先頭から実行を始めます．行の途中から実行を始めることはできません．

分岐先の前に#を付けたら，プログラムエリアへの分岐です．

GOTO #2 ……………………………P2へ分岐

IF 条件式 THEN #7 ……………………P7へ分岐

(3) プログラムエリアの分岐を実行したときの変数

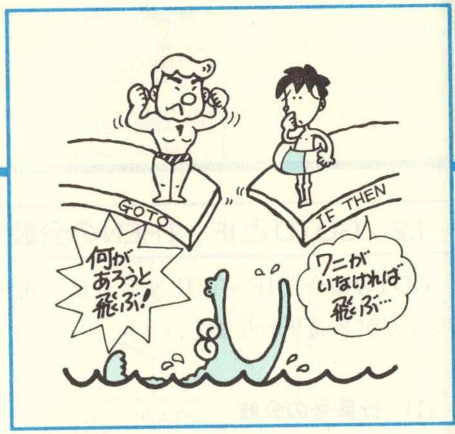
分岐を実行しても，変数の内容は変わりません．プログラムエリアが変わって，異なるプログラムを実行しても，変数の型が変わらなければ，元のまま引き続いて使えます．

No.	分岐前プログラム	分岐後プログラム	結果
1	数値変数	数値変数	データは正常に引きつがれる
2	文字変数	文字変数	
3	数値変数	文字変数	エラーは発生しない．文字変数には何も記憶されていない
4	文字変数	数値変数	分岐後，変数を実行したところでERR 6が発生

表のNo. 3，No. 4は，分岐後のプログラムで，変数の型を変えて使用した結果です．

両方とも，分岐後のプログラムで，数値変数，文字変数として代入式を実行していれば，正常に動作します．

その例を，次に示します．



〔分岐前プログラム〕

```
P0
  100  A=100:B=400
  110  Y$="ABC":Z$="500"
  120  GOTO #1
```

〔分岐後プログラム〕

P1 ………変数をそのまま使用

```
  100  PRINT "P1"                  P1
  110  PRINT "A, Y$␣";A;Y$          A, Y$␣␣100ABC
  120  PRINT "B→B$";B$             B→B$
  130  PRINT "Z$→Z";Z              ERR6   P1-130
  140  END
```

P1 ………代入式で変数の型を変更

```
  100  PRINT "P1"                  P1
```



```
  110  A$="Sunday":B$="Monday"     ………代入式で型を変更
  120  Y=10.5:Z=6                  ………代入式で変数の型を変更
  130  PRINT "A$, B$␣";A$;B$       スクロール
                                   A$, B$␣SundayMonday
  140  PRINT "Y, Z";Y;Z            Y, Z␣10.5␣6
  150  END
```

(4) **分岐先の行番号，プログラムエリアの書式**

分岐先を表わすには，次のように，数値定数，数値変数，数式を書きます．

	行番号		プログラムエリア	
数値定数	270 300.5	270行へ分岐 300行へ分岐	#0 #4.2	P0へ分岐 P4へ分岐
数値変数	N	Nの値へ分岐	#L	Lの値のプログラムエリアへ分岐
数　式	N*5	N×5の値へ分岐	#L/100	L/100の値のプログラムエリアへ分岐

* それぞれの値で小数点以下は切り捨てて実行する
* 指定する行番号が存在しない場合 ⇨ ERR4
* 指定するプログラムエリアが0より小さいか10以上の場合 ⇨ ERR2

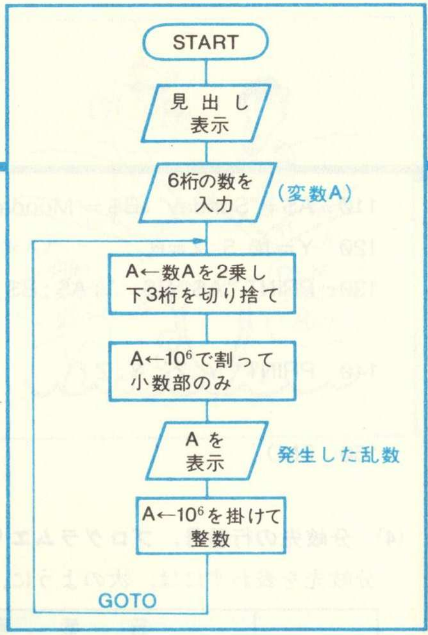
☆　☆　☆

プログラム 10 自分だけで楽しむ乱数――平方採中法

6桁の任意の数を入力して，この数を2乗します．2乗した値の下4桁目から上へ6桁取り出して，$1/10^6$すると目的の乱数が得られます．

取り出した6桁（整数）を2乗して，中央6桁を取り出すと，新たな乱数です．これを繰り返し，次々と乱数を発生させます．

■フローチャート



プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行後の表示

RANSUU	⇨ EXE
6keta no suu?	⇨ 759264（任意の数）を入力

以下の実行結果は，右の表示のようになります．

```
0.481821
0.151476
0.944978
0.98342
0.114896
0.20109
0.437188
0.133347
0.781422
0.620342
0.824196
0.299046
0.42851
```

プログラムリスト

```
10 PRINT "RANSUU"
20 INPUT "6keta no suu", A
30 A=INT (A*A/1000)
40 A=FRAC (A/1E6)
50 PRINT A
60 A=A*1E6
70 GOTO 30
```

変数　A：入力した6桁の数
　　　　　計算した乱数

プログラム解説

6桁の数として，「759264」を入力した場合で解説します．

これを2乗すると「576481821696」です．まず，下3桁を除くために1000で割り，INTで整数化します．

RNDでは四捨五入を行なうので使いません．

結果は，「576481821」となります．

この値を10^6で割り，小数部のみを取り出すと，2乗した値の下9桁目から下4桁目までの6桁の数です（0.481821）．

この値を乱数とします．

次の乱数を作るには，小数6桁に10^6を掛けて整数化した後，2乗して下3桁を取り除く行へ分岐します．

入力した値………	759264
2乗する………	576481821696
中央6桁………	481821 ……これが作った乱数
再び2乗………	232151476041

10：出力命令．見出し“RANSUU”を表示する．
20：入力命令．6桁の数をAに入力する．
30：演算．A＊AはAを2乗．1000で割って整数化（INT関数）する．
40：演算．1E6は10^6．Aを10^6で割り，小数部を取り出す（FRAC関数）．
50：出力命令．Aの値を表示する．
60：演算．Aの値に10^6を掛ける．
70：分岐命令．30行へ実行を移す．

このプログラムは，70行でGOTO命令を使っているため，STOPキーを押さない限り終了しません．

1.3　IF～THENの条件式

条件式は，2つのデータを比較することを意味しています．

データには，数字の定数，変数，式と文字の定数，文字変数を使います．

データを比較するわけですから，計算の記号があるはずです．それが関係演算記号と呼ばれる記号です．

条件式

IF　データ　関係演算記号　データ　THEN

関係演算記号は，巻末276ページの表を参照してください．

関係演算記号にも"＝"があります．これは，「左のデータと右のデータが等しいですか」という意味です．

したがって，代入文のように，「右のデータを左へ移す」という働きはしません．

このことより，条件式のデータの書き方は，左に定数，右に変数という書き方でもエラーは起こりません．

```
IF  D=-999  THEN
IF  -999=D  THEN
```

どちらも同じ処理

比較するデータは，数値変数と定数，変数式の組み合わせ，文字変数と定数，変数の組み合わせが一般に使われるので，左側に変数を書くのが普通です．

また，定数どうしの比較や，数値と文字の比較は意味がないことは，おわかりになると思います．

2　IF～；

条件に合うか，それとも……

命令　IF（条件式）；（処理）　………（条件処理）　読み方：イフ～セミコロン

2.1　IF～；は条件処理

この命令も，プログラムの中でよく使われます．

条件分岐命令のIF～THENと似た書式ですね．どちらも条件式が成立するかどうかで，処理が変わります．

ただし，IF～；の場合は，次のようになります．

条件式が成立すれば，（；）以降に書かれた命令を実行する

条件式が成立しなければ，次の行を実行する．

```
100  IF A=0;B$="GOOD":GOTO 200
110  B$="NG"
```

A=0なら → ；以降を実行 → 200行へ

A≠0なら → 110行へ

条件式は，IF～THENと同じルールです．

条件式が成立するとき実行する；以降の処理ですが，ここには，命令，式を書きます．上の例のように，いくつかの命令や式を書くこともできます．

2.2　IF～THENとIF～；の使い分け

IF～THENは，行番号またはプログラムエリアの分岐でした．

一方，IF～；は条件が成立したときのみ，（；）以降の処理を行ないます．

この2つの使い方は………

① プログラムの実行順を大きく変えるときは………THEN

② 条件が成立したときの処理を1行に書ききれないときは……THEN

③ 条件によって，わずかな処理を実行すれば，流れを変えなくてもよいときは………；



④　分岐先に入る流れが２本以上あり，条件を同じにする必要があれば………；で処理してGOTOで分岐……などがあります．

■①の例：１を選択すれば加算，２は減算，３は……というプログラムにおいて，入力した値で実行する行番号を決める場合．

```
100  INPUT "JOB:", N
110  IF N=1 THEN 1000
120  IF N=2 THEN 2000
```

■②の例：選択したキーがＹであればＡに365，Ｂに24，Ｃに…… を代入して，次の処理を実行する場合．

```
100  IF $="Y" THEN 1000
        ⁞
1000  A=365:B=24:C$="Namae": ………
1010  L=200
1020  GOTO 150
```

■③の例：プリンタへ印字するかどうかをＹまたはＮで選択する場合．

```
100  INPUT "PRTer→Y/N", $
110  MODE 8:IF $="Y";MODE 7
120     ⋮
```

■④の例：データによってクラス分けを行なう場合．

```
100  IF A>90;C$="A Class":GOTO 200
110  B=B-A
120  IF A>70;C$="B Class":GOTO 200
```


2.3　IF命令を２つ以上重ねる

IF～；とIF～THEN，またはIF～；とIF～；を２つ以上１文に書くこともできます．

この場合，条件式は前に書かれている方から順に，比較を行ないます．

```
100 IF A$="Y";              IF B>5;P$="PASS"
A$="Y"のとき ─────────────→
A$≠"Y"のとき ─┐    A$="Y"で，B＞５のとき →
              │    A$="Y"で，B≦５のとき ┐
              └──────────────→ 110 …………
```

条件 A$	処　理	条件 B	処　理
Y	条件Ｂへ分岐	５を越える	P$="PASS"（実行後110行へ分岐）
		５以下	110行へ分岐
Ｙ以外	110行へ分岐		

上記のように，途中で成立しない条件式があったら，その行はそれ以上実行せず，次の行へ移ります．

2.4　文字の大小比較──文字はアスキーコードで大小を決める

①　文字を２つ入力して大小を表示

```
10  PRINT "MOJI no Hikaku"
20  INPUT "MOJI 1:", A$, "MOJI 2:", B$
30  IF A$>B$;C$=">"
40  IF A$=B$;C$="="
50  IF A$<B$;C$="<"
60  PRINT A$;C$;B$
70  GOTO 20
```

（＊286ページのキャラクター表を参照）

上記のプログラムは，２つの文字列の大小を比較するものです．

プログラムでは，入力した文字の大小関係を表わすために，条件式が成立

すれば，比較演算記号と同じ記号を変数C$に代入しています．

1つの条件文を実行した後は，条件が成立している，いないにかかわらず，次の行の条件文を実行しています．

たとえば，リスト(A)のように，30行ではA$がB$より大きくても，C$に＞を代入した後は，40行を実行します．

A$とB$の間では，3つの関係しかありません．つまり，大きいか，等しいか，小さいかです．

したがって，30行実行後に，40行と50行を実行しても，条件が成立するのは1回だけです．

30行と40行にGOTOを使えば，リスト(B)のように書くこともできます．50行はA$＜B$だけなので，条件文を使わないことに注意してください．

リスト(A) （48ステップ）

```
30 IF A$>B$;C$=">"
40 IF A$=B$;C$="="
50 IF A$<B$;C$="<"
```

⇕ 同じ機能

リスト(B) （49ステップ）

```
30 IF A$>B$;C$=">":GOTO 60
40 IF A$=B$;C$="=":GOTO 60
50 C$="<"
```

② 一番大きい文字，小さい文字

```
10 L$="␣":S$="■"
20 PRINT "MOJI Hikaku"
30 INPUT "MOJI:", X$
40 $=L$:Y$="Max."
50 GOSUB 500
60 IF C$="<";L$=X$
70 $=S$:Y$="Min."
80 GOSUB 500
90 IF C$=">";S$=X$
100 GOTO 30
500 IF $>X$;C$=">"
510 IF $=X$;C$="="
520 IF $<X$;C$="<"
530 PRINT Y$;"→";$;C$;X$
540 RETURN
```


一番大きい文字と小さい文字を探し出すこのプログラムでは，文字列の比較に①と同じ方法を使っていますが，サブルーチンとして，最大にも最小にも利用できるようになっています．

入力した文字が最大，最小の文字と入れ替わるのは，サブルーチンで代入した比較演算記号によります．

変数C$には，＞，＝，＜の1つが代入されているので，最大では＜なら入れ換え，最小では＞なら入れ換えます．変数と決まった文字定数の比較で条件が作られているわけです．

なお，文字列の大小は，巻末286ページのキャラクター表の順です．

2.5 IF～;GOTOはIF～THENと同じ

```
100 IF A=0;GOTO 600  ⇔  100 IF A=0 THEN 600
```

同じ働きをする

左右の2つの処理は全く同じです．

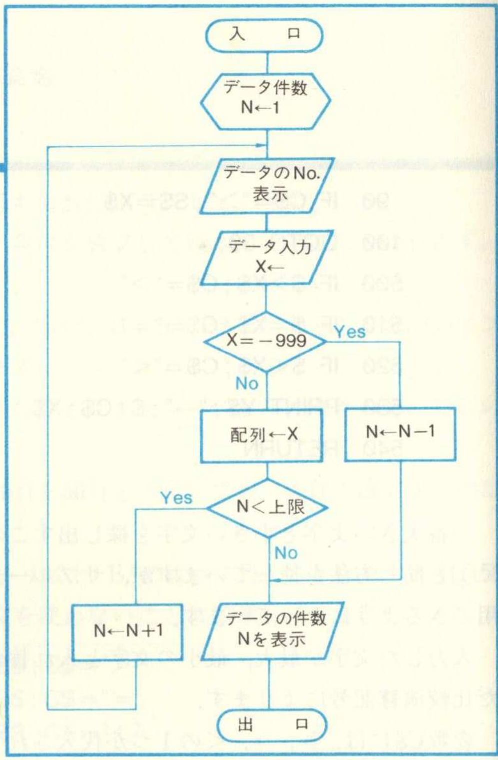
条件式A＝0が成立すれば，共に600行へ分岐します．つまり，条件分岐命令のTHENは，条件文の ;GOTO と同じ意味となります．いろいろな命令を覚えると，ウッカリして，次のようなムダな書き方をする場合があります．

```
IF 条件式;N=20:GOTO N
```

これでも，PB-300は正常に動作処理しますが，複雑になるだけです．

簡単に，THEN 20 ですみます．

■フローチャート



プログラム　11

データ数の不明なデータを入力するサブルーチン

データを入力する場合に，前もってデータ数が不明なとき，入力終了後にデータ数が何件であったか知りたいことがあります．

次のプログラムは，そのようなときに役立つデータ入力のサブルーチンです．

プログラム入力準備

[MODE] 1
CLEAR [EXE]
DEFM 20 [EXE]
（メモリー容量に合わせて設定してください）

プログラム実行

[MODE] 0
RUN [EXE]
（メインルーチンより）

実行後の表示

表示	操作
No. 1?	⇨ 21 を入力
⋮	
No. 11?	⇨ −999 を入力
Data　10　ken	⇨ [EXE]を押すとメインルーチンへ戻る

プログラムリスト

```
1000  N=1
1010  PRINT "No.";N;:INPUT X
1020  IF X=-999;N=N-1:GOTO 1040
1030  Z(N)=X:IF N<G;N=N+1:GOTO 1010
1040  PRINT "Data";N;"ken"
1050  RETURN
```

変数　G：データ数の上限，メインルーチンで設定
　　　N：データの件数
　　　X：入力したデータ
　　　Z(N)：データを記憶しておく配列

プログラム解説

このサブルーチンでは，データを配列へ記憶しているので，実行時にはDEFMでメモリーの拡張が必要です．メインルーチンでは，拡張した配列の大きさに合わせて，データ件数の上限をGに設定します．

データ入力の終了は，−999を入力するか，データの件数が配列の上限に達したときです．データの入力が終ったら，入力した件数を表示してメインルーチンへ戻ります．

1000：データの件数を1に設定する．変数はN．
1010：データのNo.を表示して，入力を受け付ける．データはXへ入力．
1020：入力されたデータが −999 だったら，データの件数を −1 して，1040行へ分岐する．データ入力の終了．
1030：入力したデータを配列に代入する．その後で，データの件数が上限以上（データ入力終了）か，上限より小さい（データ件数に＋1）かを判定する．
1040：入力したデータの件数を表示する．
1050：サブルーチンの終了．メインルーチンへ戻る．

3　GOSUB, RETURN
サブルーチン

2つのコマンドを必ずペアで使います．

命令		読み方
	GOSUB（行番号またはプログラムエリア）	：ゴーサブ
	RETURN	：リターン

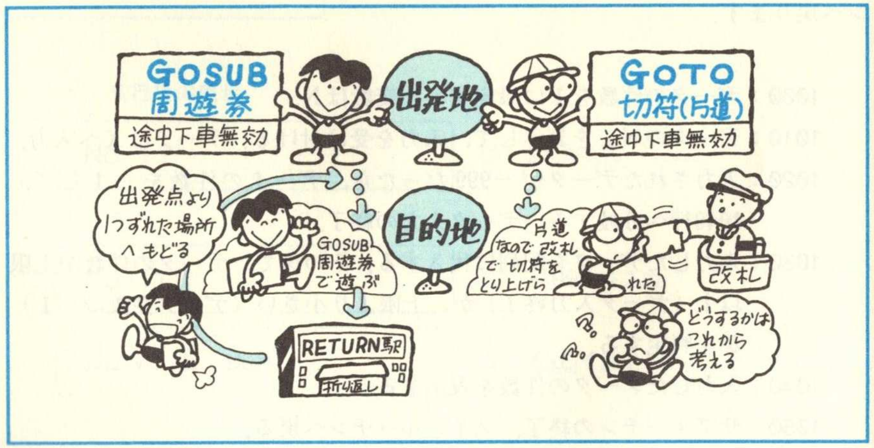
3.1　GOSUBとGOTOの違い

GOSUBもGOTOも，行番号，プログラムエリアを指定し，両方とも，指定された行，プログラムエリアへ無条件に分岐していくことは同じです．

そして，行った先で，プログラムに書かれていることを処理するのも同じです．

では，どこが違うのでしょうか．

プログラムの流れからいえば，GOSUBで分岐したプログラムの流れには，必ず区切りとなる命令があることです．一方，GOTO命令でのプログラムの流れは，行き先まかせであることです．この点が違うわけです．





命令の実行状態からいえば，RETURN命令に出合ったときホッとするか，いやなヤツに会ったと思うかの違いです．

GOSUB命令では，RETURNを実行すると，命令の実行を終了して，GOSUBを実行した行の次の命令，または次の行へ戻ります．

GOTO命令で分岐して，RETURNを実行すると，意味不明な命令として，エラー（ERR 7）を表示します．

3.2　サブルーチンとメインルーチン

GOSUBで分岐して実行するプログラムのことを，サブルーチンと呼びます．サブルーチンは，GOSUBで指定される行番号から，RETURNではさまれます．

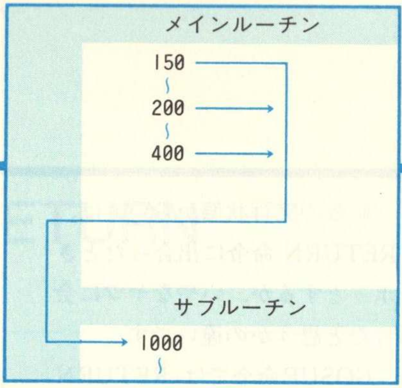
サブルーチンに対して，処理の大きな流れを書いたプログラムのことを，メインルーチンと呼びます．

サブルーチンには，メインルーチンで何回も使うため，処理のひとまとまりを書いてプログラムをスッキリさせる場合と，1回しか実行しないが，プログラムを構造化するため，ブロックごとにまとめて，サブルーチンにする場合があります．

① プログラムの中で何回も使う

次のページのプログラム形式は，メインルーチンで何回もサブルーチンを利用する例です．たとえば，プリンタを使用して，何度も横線を印字するとします．そこで，プリンタで横線を引くプログラムをサブルーチンとします．

このプログラムでは，横線を引く必要が発生するのは，150行，200行と




```
    ⌇
150 GOSUB 1000
    ⌇
200 GOSUB 1000
    ⌇
400 GOSUB 1000
    ⌇
1000 IF $="Y"; PRINT "…………………………"
1010 RETURN
```

400行です．

横線を印字するかどうかは，1000行で決めます．

メインルーチンのGOSUB命令によって，一時的に1000行の処理へ実行を移します．1000行の処理が終われば，メインルーチンの次の命令に戻り，メインルーチンの処理を継続します．

1つの処理が何回も発生する場合，このように，その処理をサブルーチンとしてまとめておけば，プログラムのステップ数も短くでき，見やすいプログラムを書くことができます．

② プログラムの構造化にサブルーチンを使う

次のページのプログラム形式は，前述したような，同じ処理を何度も使用するためにサブルーチンとしたというわけではありません．

1回しか利用しない処理も，すべてサブルーチンとしています．

そして，メインルーチンの処理は，サブルーチンのどこを実行するとか，初期値の設定を行なうなどの基本的な事がらだけに限定しています．

つまり，処理の詳細は，各サブルーチンで行なう形式になっています．

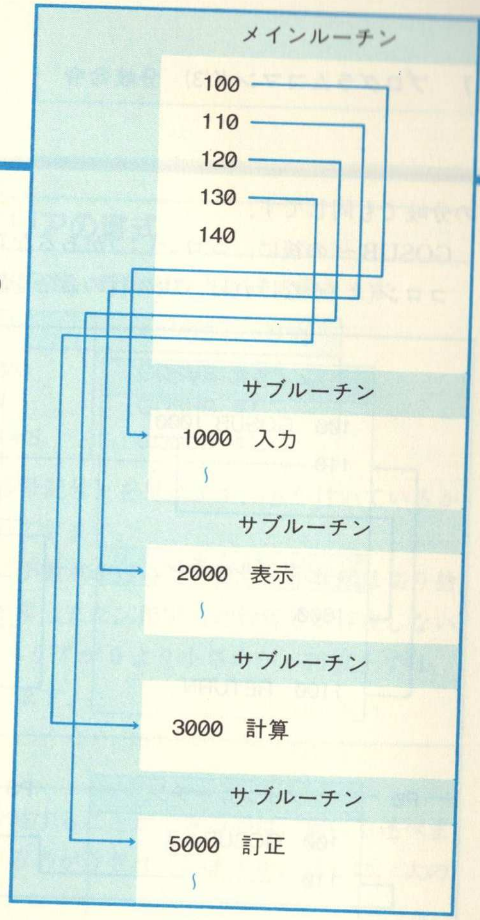
もし，この中で共通する処理があれば，①の考え方で，サブルーチンを作ります．また，サブルーチンからサブルーチンを実行するのは8回まで


```
100 GOSUB 1000
110 GOSUB 2000
120 IF $="Y"; GOSUB 5000
130 GOSUB 3000
140 END

1000 ────
      ⌇
1200 RETURN
2000 ────
      ⌇
2100 RETURN

3000 ────
      ⌇
3100 RETURN

5000 ────
      ⌇
5200 RETURN
```



許されます．

プログラムエリアを10個持つPB-300では，この構造化したプログラムが効果を発揮します．

3.3 GOSUBの分岐とRETURN

GOSUB命令では，行とプログラムエリアの分岐を行ないます．

行番号の指定では，指定した行から実行を始めます．これは，同じプログラムエリア内に限られます．

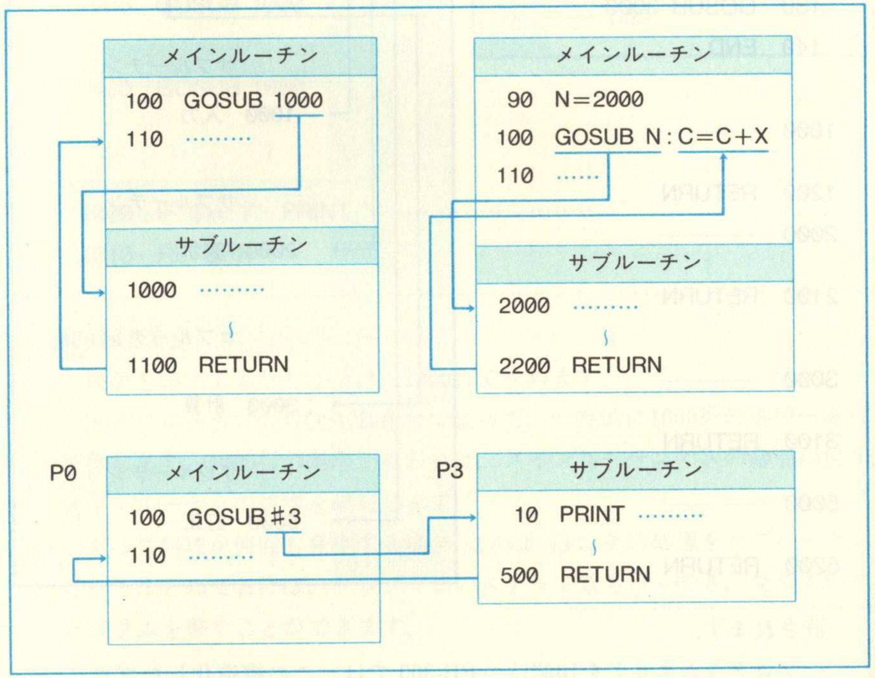
プログラムエリアの指定は，指定したプログラムエリアの先頭の行から実行を始めます．

RETURN命令を実行したときの戻り先は，行の分岐でもプログラムエリア

の分岐でも同じです．

GOSUB～の後に，コロン(：)があるなら，：以後の命令を実行します．

コロン(：)がなければ，次の行の命令を実行します．



3.4 プログラムエリア間のサブルーチンを実行したときの変数

メインルーチンからサブルーチン，サブルーチンからメインルーチンへのデータは，そのまま使えます．

ただし，次のように，GOTO命令と同じ制限が加わるので注意してください．

① 変数の型が変わらなければ，記憶している内容はそのまま使える．

② 変数の型を変えた場合，代入式を実行してからでないと，正常な値が記憶されていないか，エラーが発生する．

3.5 行番号，プログラムエリアの書式

行番号とプログラムエリアは，数値定数，数値変数，数式で書きます．

	行　番　号	プログラムエリア
数値定数	GOSUB 200	GOSUB ♯4
数値変数	GOSUB N	GOSUB ♯L
数　　式	GOSUB N＊5	GOSUB ♯L/10

プログラムエリアの指定には♯（番号記号）を付けます．♯が付いているか付いていないかで，意味が大きく違ってきます．

行番号，プログラムエリアの指定に小数点が付いていたら，小数部は切り捨ててサブルーチンの分岐先と判定します．また，指定した行番号が存在しないときはERR 4，指定したプログラムエリアが0より小さいか，10以上では，ERR 2を表示してプログラムは中断します．

3.6 GOSUBは8回まで

サブルーチンからサブルーチンへ分岐することができるのは，前にも述べました．8回までは，GOSUBで分岐するのが許されていますが，試しに，次のプログラムで確かめてみましょう．

```
100  N=1
1000 PRINT CSR 0;N;"Kaime Try";:FOR I=1 TO 200:
     NEXT I
1010 GOSUB 2000
2000 PRINT CSR 0;N;"Kaime OK!";:FOR I=1 TO 200:
     NEXT I
2010 N=N+1
2020 GOTO 1000
```



1回目のメインルーチン

100：Nに1を代入
1000：1回目のGOSUB
　　を実行する表示
1010：GOSUB 2000
　　を実行

2回目からのひとまとまり

2000：N回目のGOSUB
　　を実行したことを表示
2010：Nに1を足す
2020：1000行へ分岐（GOTO命令）
1000：2回目のGOSUBを実行することを表示
1010：GOSUB 2000を実行
2000：
　⋮

このプログラムを実行すると，1 kaime Try 1 kaime OK!……… 9 kaime Try と順に表示されます．

ところが，9回目のGOSUBに挑戦したところで，ルールどおりエラーが発生します．表示は，〈ERR7 P0-1010（P0で実行した場合）〉となります．

このプログラム例では，GOSUBを何回も重ねるために，わざと2020行から1000行へ無条件分岐させています．

プログラムを作る段階で，サブルーチンの範囲はここまでであると，区切りをはっきりさせておかないと，GOSUBで分岐したのに，いつのまにかサブルーチンの外へ飛び出していた，ということが起こりかねません．

サブルーチンからRETURN以外で飛び出したとしても，そのときPB-300は，あなたに何も教えてくれません．

実行を重ねて行くうちに，突然エラーが発生したり，とんでもないところへ戻っているなんてことになります．

そうなると，原因究明（デバッグ）は大変です．プログラムを作る段階で是非つぶしておいてください．

プログラム 12　JOB MENUの表示方法と入力方法

プログラムの中で行なう仕事（JOB）を選択入力する場合に，12文字以内では，すべてのメッセージを表示できません．

そこで，JOBの種類を1個ずつ順に表示させながら，いつでも選択入力ができるプログラムを作ってみます．

サンプルとして，ここでのJOBは2種類ですが，同じ書式で何個でも増やせます．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行後の表示

＊GOSUB　TEST＊
1：Kasan
2：Hikizan
＊GOSUB　TEST＊
2：Hikizan

（＊GOSUB　TEST＊）………一定時間表示後，自動的に次の表示に変わる．その間，いつでもキー入力ができる．

（＊GOSUB　TEST＊）⇨　2を入力

（2：Hikizan）………300行を実行

プログラムリスト

メインルーチン

```
100  GOSUB 1000
110  IF K$="1" THEN 200
120  IF K$="2" THEN 300
130  GOTO 100
```

JOB1

```
200  PRINT "1:Kasan"
210  PRINT "END":END
```

JOB2

```
300  PRINT "2:Hikizan"
310  PRINT "END":END
```

表示サブルーチン

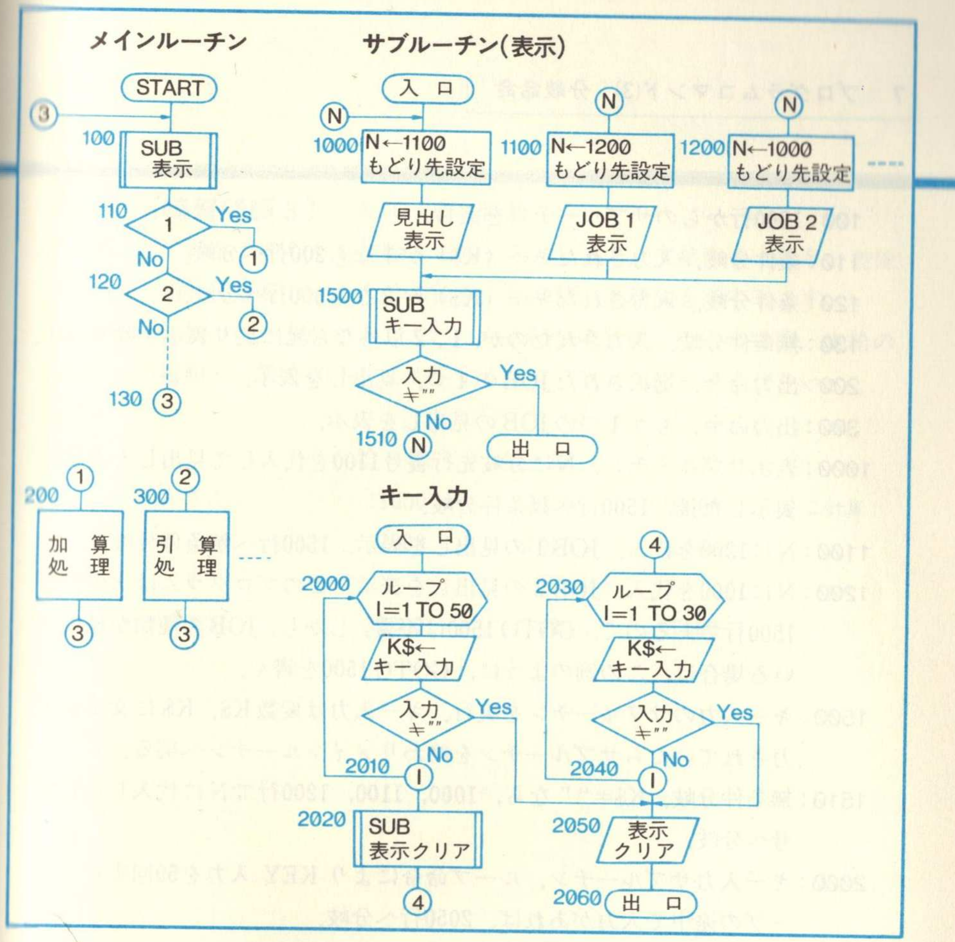
```
1000  N=1100:PRINT "*GOSUB TEST*";:GOTO 1500
1100  N=1200:PRINT "1:Kasan";:GOTO 1500
1200  N=1000:PRINT "2:Hikizan";:GOTO 1500
1500  GOSUB 2000:IF K$≠"";RETURN
1510  GOTO N
```

キー入力サブルーチン

```
2000  FOR I=1 TO 50:K$=KEY:IF K$≠"" THEN 2050
2010  NEXT I
2020  GOSUB 2050
2030  FOR I=1 TO 30:K$=KEY:IF K$≠"" THEN 2050
2040  NEXT I
2050  PRINT CSR 0;"␣␣␣␣␣␣␣␣␣␣␣␣";:PRINT
      CSR 0;
2060  RETURN
```

変数 I:ループカウンタ
K$:キー入力した文字1字
N:分岐先行番号

■フローチャート



プログラム解説

このプログラムは，メインルーチンと2つのサブルーチンから構成します．

メインルーチンは，選択されたキーによる分岐先を決める条件分岐と目的の処理です．2つのサブルーチンは，表示とキー入力です．

表示サブルーチンは，変数名で分岐先を指定できる特徴を生かして，戻り先の設定と表示，キー入力があったかどうかの条件文です．この中で，自らのサブルーチン内のプログラムを，サブルーチンとして使っています．1つのRETURN命令をいくつかのGOSUB命令で使うことができます．

表示するJOBの種類を増やすには，メインルーチン100番台の条件分岐とサブルーチン表示の1000～1200の書き方をペアで増やします．

100：1000行からのサブルーチンを実行．
110：条件分岐．入力されたキー（K$）が1なら200行へ分岐．
120：条件分岐．入力されたキー（K$）が2なら300行へ分岐．
130：無条件分岐．入力されたのが，1，2以外なら元に戻り表示を繰り返す．
200：出力命令．選択されたJOBの1つ．見出しを表示．
300：出力命令．もう1つのJOBの見出しを表示．
1000：表示サブルーチン．Nに分岐先行番号1100を代入して見出しを表示．表示した後，1500行へ無条件分岐．
1100：Nに1200を代入．JOB 1の見出しを表示．1500行へ無条件分岐．
1200：Nに1000を代入．JOB 2の見出しを表示．このプログラムは次の行が1500行であるので，GOTO 1500は不要．しかし，JOBの種類が続いている場合は，この例のように，GOTO 1500を書く．
1500：キー入力のサブルーチンを実行．キー入力は変数K$．K$に文字が入力されていたらサブルーチンを終わりメインルーチンへ戻る．
1510：無条件分岐．K$≠“”なら，1000，1100，1200行でNに代入した行番号へ分岐．
2000：キー入力サブルーチン．ループ命令によりKEY入力を50回実行．ループの途中で入力があれば，2050行へ分岐．
2010：2000行のループ命令と対．ループの折り返し．
2020：2050行から2060行のRETURNまで実行するGOSUB命令．GOSUBとRETURNが対であれば，このような実行も可能．処理は表示のクリア．
2030：ループ命令でKEY入力を実行．今度は，表示がクリアされたままでキー入力を実行．その分はループの回数も少なくする．ここでキーが押されたら2050行へ条件分岐．
2040：2030行のループ命令と対．
2050：表示をクリアし，次のPRINT命令の表示位置を先頭に設定．
2060：サブルーチンの出口．戻り先は1500行のIF～と2030行の2つがある．GOSUBを実行した行番号により戻り先も決まる．

〔練習問題3〕 たて，よこ計算

給与と給与から控除する額を入力します．支給額を（給与－控除額）で計算し，1人分の給与，控除額，支給額を印字します．

給与に－999が入力されたら，給与の総額，控除の総額，支給の総額，人数を印字して終了するプログラムを作ってください．

	給与	控除	支給	
Aさん	11000	1000	10000	⇦よこ計算
Bさん	7000	2000	5000	
﹏	﹏	﹏	﹏	
たて計算⇨	82000	15000	67000	

答えは226ページ⇨

8 プログラムコマンド(4) ループ命令

1 FOR・NEXT

決めた数だけ繰り返す

2つのコマンドは，必ずペアで使います．

命令	
	FOR 変数＝初期値 TO 終値〔STEP 増分〕 ～ 処　理 ～ NEXT 変数

〔STEP 増分〕は省略できる

読み方：フォー～イコール～トゥー～ステップ
～
ネクスト

決められた回数だけ，FORとNEXTにはさまれた処理を繰り返し実行する命令です．

また，処理が何も書かれていない場合は，終値まで数を数えるという処理を行ないます．FORとNEXTは必ずペアで，FORを先にNEXTは後に書きます．

1.1　変数

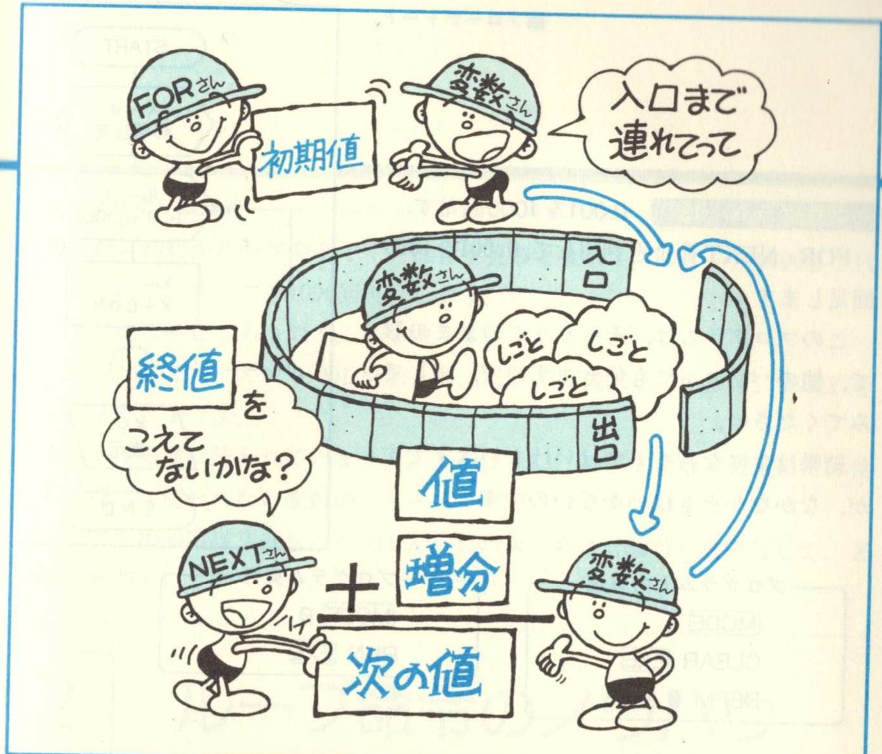
変数には，AからZの数値変数を使います．A(1)などの配列は使えません．

この変数に初期値から終値までを増分置きに代入して，FORとNEXTにはさまれている処理を実行します．

PB-300は，FORの次に書く変数と，NEXTの次に書く変数は同じもので，ペアで処理します．

1.2　初期値

初期値には，PB-300で演算できる範囲の数値定数，変数，数式を書きます．



FOR命令を実行すると，変数にはまず初期値が代入され，ループの中の処理が行なわれます．

1.3　終値

終値に書けるのは，PB-300で演算できる範囲の数値定数，変数，数式です．

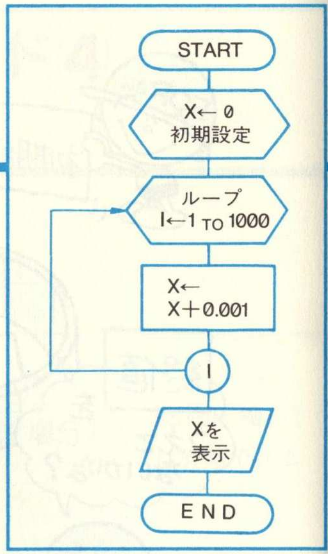
FOR命令を実行するときの変数に代入する値の上限，または下限を終値で設定します．終値を越えない範囲で，命令は実行されます．増分の設定によっては，終値は変数に代入されないこともあります．

1.4　増分

〔STEP 増分〕は省略してもかまいません．そのときは〔STEP 1〕（1ずつ増える）とみなされます．増分に書ける数値も同じく，PB-300で扱える範囲の数値定数，変数，数式です．小数や負の数も書けます．

増分に書いた値をループ1回ごとに初期値に加えていきます．増分が負なら引いていきます．引いたり，加えたりして，終値を越えたらループの終りです．

■フローチャート



プログラム 13　0.001を1000回足す

FOR・NEXT命令を利用して，0.001を1000回足します．

このプログラムは，そっくりこのままの形で，他のパソコンにも使えますので，試してみてください．

結果は1にならなくてはいけないはずですが，なかなかそうはいかないのですよ．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

（空白）　RUN入力後，しばらく何も表示しません．

1

プログラムリスト

```
10 X=0
20 FOR I=1 TO 1000
30 X=X+0.001
40 NEXT I
50 PRINT X
60 END
```

変数　I：ループカウンタ
　　　X：加算結果

プログラム解説

20行から40行がループ命令です．20行を見ると変数Iには1から1000まで代入されるので，ループを1000回実行します．

STEPは省略してあるので，1です．

処理は30行です．Xに0.001を加えて，再びXに代入しています．0.001を1000回足すと1です．

変数Iは処理に使われていません．このように，ループ命令の変数にだけ使い，処理には使われない変数のことをループカウンタと呼びます．

ループを1000回繰り返したらNEXT Iの次の命令（50行）を実行して，Xの値1を表示します．

2　ループ命令のノウハウ

2.1　注意したいSTEPの使い方

よくやるミスとして……

FOR I=100 TO 20 STEP 1

……といった例があります．

コーディングする際に，1ずつ変わるのだから，STEP 1と書いてしまうのですが，初期値と終値を見ればわかるように，100から20まで減らしながらループ命令を実行させようとしているのです．

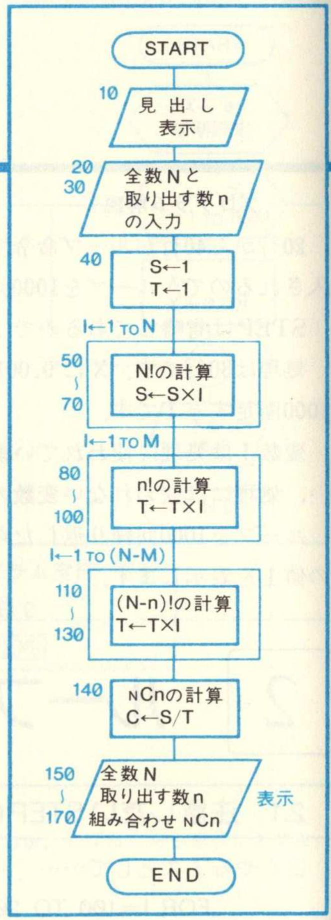
このままですと，ループ命令は変数Iに100を代入して，NEXT Iまでの処理を1回だけは実行しますが，2回目は変数Iに101が代入され，終値を越えるため実行しません．プログラムをRUNして，初めてこのミスに気が付くことになるわけです．正しくは……

FOR I=100 TO 20 STEP −1

……です．これで，100から1ずつ減らして20までループを実行します．

1ずつ加えると，1ずつ減らすとは大変大きな違いです．

■フローチャート



プログラム 14　組み合わせ $_NC_n$ の数

このプログラムは，N個の中からn個を取り出す組み合わせは何組かを計算します．

次の計算式を見ておわかりのように，階乗の計算ですから，あまり大きい数を与えると演算範囲を越えてエラーが発生します．

計算式　$_NC_n=\dfrac{N!}{n!(N-n)!}$

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

	表示	キー入力
	＊KUMIAWASE＊	⇨ EXE
全数Nの入力	Zensuu N : ?	⇨ 40 EXE
取り出す数nの入力	スクロール Toridasu suu n : ?	⇨ 5 EXE
入力した全数	Zensuu N= 40	⇨ EXE
入力した取り出す数	スクロール Toridasu suu n= 5	⇨ EXE
組み合わせ数	スクロール Kumiawase C= 658008	⇨ EXE プログラム終了

プログラムリスト

```
10 PRINT "＊KUMIAWASE＊"
20 INPUT "Zensuu N:", N:N=INT (N)
30 INPUT "Toridasu suu n:", M:M=INT (M)
40 S=1:T=1
50 FOR I=1 TO N
60 S=S*I
70 NEXT I
80 FOR I=1 TO M
90 T=T*I
100 NEXT I
110 FOR I=1 TO (N-M)
120 T=T*I
130 NEXT I
140 C=S/T
150 PRINT "Zensuu N=";N
160 PRINT "Toridasu suu n=";M
170 PRINT "Kumiawase C=";C
180 END
```

変数	
C	組み合わせ数，$_NC_n$
I	ループ変数，階乗の掛ける数
M	取り出す数n
N	全数N
S	N! の計算結果
T	n!(N−n)! の計算結果



──プログラム解説──

5の階乗は，5！＝1×2×3×4×5　です．ループ命令で計算するには，ちょうどいい例題です．

50～70行ではN！を，80～100行ではn！を，110～130行では(N－n)！を計算します．

ループ命令の変数は，処理の中では掛ける数です．掛け算ですから，計算した結果を記憶する変数の初期値は1に設定しています．

入力する個数に小数点は考えられないので，整数化（INT関数）します．

10：見出しを表示する．

20：全数Nを変数Nに入力する．入力した値はINTで整数化する．

30：取り出す数nを変数Mに入力する．MはINTで整数化する．

40：階乗の計算結果の変数を初期値1に設定する．

50～70：Nの階乗を計算するループ命令．1からNまでをSに掛けて，結果をSに記憶する．

80～100：nの階乗を計算するループ命令．1からn（変数M）までをTに掛けて，結果をTに記憶する．

110～130：(N－n）の階乗を計算する．結果はTに記憶する．

140：組み合わせ数$_NC_n$を計算する．変数C

150：全数Nを表示する．

160：取り出す数nを表示する．

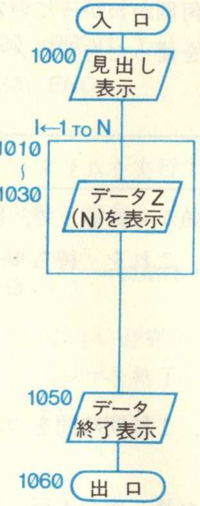
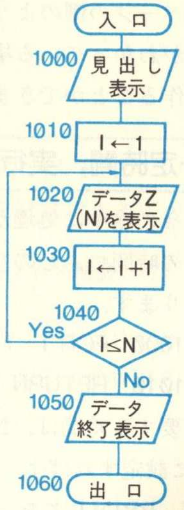
170：組み合わせ数$_NC_n$を表示する．

180：END文．プログラムの終了

2.2　分岐命令でループを作る

GOTOとIF～THENを組み合わせることで，FOR・NEXT命令と同じ働きをさせることができます．

例として，配列Z(1)～Z(20)に入っているデータN件を表示させるサブルーチンを，分岐命令とループ命令の両方で作ります．

ループ命令	分岐命令
1000 PRINT "Data PRT" 1010 FOR I=1 TO N 1020 PRINT "No.";I;":";Z(I) 1030 NEXT I 1050 PRINT "Data END" 1060 RETURN	1000 PRINT "Data PRT" 1010 I=1 1020 PRINT "No.";I;":";Z(I) 1030 I=I+1 1040 IF I≦N THEN 1020 1050 PRINT "Data END" 1060 RETURN
	

実行時の表示

Data　PRT
No. 1:　62
⋮
Data　END

表示は，ループ命令，分岐命令ともに同じである

分岐命令により，ループ命令と同じ処理を行なうには，1010行の変数の初期値設定，1030行の増分だけの加算または減算（ここでは＋1が増分），1040行の終値を越えないかの判定と分岐を独立して書きます．

データ数の不明なデータには，分岐命令を使う方がエラーの発生しないプログラムを作れます．このような例として，〈プログラム 11〉があります．

一方，前ページの例のように，データの件数が何個かわかっている場合や，数値の範囲がわかっている場合には，ループ命令を使った方が省メモリーのプログラムを作ることができます．

2.3　一定時間，実行を中断する

ループ命令の途中に処理が何もなければ，初期値から終値まで，数をカウントするだけの時間つぶしのループができ上ります．これを「待ちループ」と呼ぶこともあります．

```
1000  FOR I=1 TO N:NEXT I
1010  RETURN
```

ループに要する時間は，N＝100で約0.7秒です．１秒間時間をつぶすにはNを140前後に設定すればよいことになります．

たとえば，PRINT文を一定時間だけ表示させた後，次の実行に移る場合や，ゲームのプログラムに利用すると便利です．

次の例は，このFOR・NEXT命令を使って，cos 0°からcos 90°までを一定時間ずつ次々に表示するプログラムです．

```
100  PRINT "* COS *"
110  MODE 4
120  FOR I=0 TO 90 STEP 15
130  SET N:PRINT I;"°=";
140  SET F3:PRINT COS I;
150  N=200:GOSUB 1000
160  NEXT I
170  END
1000 FOR J=1 TO N:NEXT J
1010 RETURN
```

2.4　FOR命令終了時の変数の値はどうなっているか？

```
100  FOR I=1 TO 5 STEP .5   ……0.5は.5でもよい
110  NEXT I
120  PRINT "I=";I
130  END
```

上記のプログラムを実行すると，I=5.5 と表示されます．

ループ命令終了時のループ変数の値は，次の式で計算できます．

$$\text{ループ終了時の変数の値} = \text{初期値} + \left\{ \text{INT}\left(\frac{\text{終値}-\text{初期値}}{\text{増　分}} \right) + 1 \right\} \times \text{増分}$$

しかし，このように計算はできますが，ループ変数はループ内だけで有効であると考えて，ループを終了したら，その変数の値をそのまま使うのはお推めできません．

2.5　FOR命令の処理途中に変数の値を変えると？

```
100 FOR I=1 TO 10
110 PRINT "START I=";I
120 INPUT "Hikisuu:", H
130 I=I-H
140 PRINT "CAL I=";I
150 NEXT I
160 END
```

実行時の表示　　キー入力

表示	キー入力
START I=1	
Hikisuu : ?	⇨ 3 EXE
CAL I=−2	
START I=−1	
START I= 3	
Hikisuu : ?	⇨ 1 EXE
CAL I= 2	

プログラム実行直後に3を入力すると，1−3=−2です．

次のループでは増分だけ加えて，−1です．初期値よりも小さくなりました．

START I=3を表示したところで，1を引いてみます．次もI=3でループしました．今度は2を引いてみます．次のループは，計算した値+1（増分）がIに代入されて実行します．

表示	キー入力
START I= 3	EXE
Hikisuu : ?	⇨ 2 EXE
CAL I= 1	EXE
START I= 2	⇨ 未記入は EXE を押す

これより，ループ命令の変数は，

① 初期値より小さくもできる．

② 処理中に+かーをしたら，計算した変数の値+増分で，次のループを始める．

……という性質をもちます．

プログラムのステップ数が少ない場合は，変数の値を変えても人間が管理できますが，大きなプログラムの中でループ命令の変数を変えると，思わぬエラーを起こすことがあります．

PB-300はプログラム通りに実行しているのですから，プログラムを作る段階のミスということになります．この種のエラーはデバッグがやっかいで，原因を発見するまでに時間がかかります．

プログラムの中で変数の値を変えることはできますが，お推めしません．他の方法で処理できるように工夫してください．

2.6　FOR命令の途中に分岐命令が飛び込むと？

```
100 I=5
110 INPUT "Jump→Y/N", $
120 IF $="Y" THEN 140
130 FOR I=1 TO 10
140 PRINT "I=";I
150 NEXT I
160 END
```

……Yを入力したら140行へ分岐する．このときIは5に設定している．はたして結果は？

ループ（130〜150行）

実行時の表示

表示	キー入力
Jump → Y/N ?	⇨ Y EXE
I= 5	……140行を実行してIの値を表示
ERR 7 P0-150	……NEXT I を実行したところでエラー

Jump→Y/N で，Y を入力すると，120行から140行へ分岐します．

そして140行は実行しました．ところが，150行を実行したところで ERR 7 のエラーが発生します．このように，ループに途中から飛び込むことは許されません．

FOR命令とNEXTは常にペアでしたね．ちょうど，GOSUBとRETURNの関係に似ています．変数に数値を代入してあろうが，なかろうが，関係なく戻っていく先がわからないとERR 7 を発生します．

飛び先の行番号を付け間違うと，このような結果になります．

2.7 FOR命令の処理の途中から分岐命令で飛び出すと？

```
100 FOR I=1 TO 10
110 PRINT "I=";I
120 INPUT "Jump→Y/N", $
130 IF $="Y" THEN 150   ……Yが入力されたら ループの外へ飛び出す
140 NEXT I
150 PRINT "Jump I=";I
160 END
```

（100〜140行：ループ）

ループの途中から分岐すると，何の支障もなく，ループ内の処理結果を引き続き使用できます．ループ内で計算した値でも，ループ命令の変数も使用してかまいません．ただし，FORネスティングレベルは，NEXT文かVAC でクリアしない限り，残った（使われた）ままになっています．



ループ命令終了時の変数は使わない方がよい，といいましたが，途中から飛び出す場合は，使ってもさしつかえありません．

実行時の表示

表示	キー入力
I= 1	⇨ EXE
⋮	
I= 5	⇨ EXE
Jump→Y/N ?	⇨ Y EXE
Jump I= 5	……150行を実行

2.8 FOR・NEXT命令のネスティング

GOSUB命令では，8重までのネスティングができることを述べましたが，FOR・NEXT命令では，次のように4重まで許されます．

```
FOR A=…… TO ……
 FOR B=…… TO ……
  FOR C=…… TO ……
   FOR D=…… TO ……
      ⌇
     処 理
      ⌇
   NEXT D
  NEXT C
 NEXT B
NEXT A
```

FOR・NEXT命令のネスティングで注意することは，FORに対応するNEXTの変数どうしが，前述のような位置関係をもたねばならないことです．

たとえば，次の例のような組み方をすることはできません．

```
┌FOR A=…… TO ……
│┌FOR B=…… TO ……
││
└┼NEXT A
 └NEXT B
```

FOR・NEXT命令のネスティングは4重まで許されるといいましたが，はたして，それ以上はどうなるか，次のプログラムで試してみます．

```
10  N=0:GOSUB 300
20  FOR A=1 TO 10:GOSUB 200
30  FOR B=1 TO 10:GOSUB 200
40  FOR C=1 TO 10:GOSUB 200
50  FOR D=1 TO 10:GOSUB 200
60  FOR E=1 TO 10:GOSUB 200
70  FOR F=1 TO 10:GOSUB 200

100 END
200 PRINT N;"␣Kaime␣OK!"
300 N=N+1
310 PRINT N;"␣Kaime␣TRY"
320 RETURN
```

実行時の表示

表示	
1 Kaime TRY	⇨ EXE

表示	
1 Kaime OK!	⇨ EXE
2 Kaime TRY	⇨ EXE
2 Kaime OK!	⇨ EXE
4 Kaime TRY	⇨ EXE
4 Kaime OK!	⇨ EXE
5 Kaime TRY	⇨ EXE
ERR7 P0-60	……5回目のトライでエラー発生

このプログラムを実行し，エラーが出たところで，マニュアルモードで，A B, C, D, Eの変数の内容を調べてみますと，A~Dは1ですが，Eは0となっています．

つまり，60行を実行するときにエラーとなり，Eが初期値1に設定されていないわけです．

なお，FOR・NEXTループからの飛び出しを1回実行すると，ネスティングは1段使われたままになります．

☞ 〔**練習問題 4**〕　コード別集計

区分コード，数量，単価を入力します．

区分コードは1から9までに分かれ，99が入力されたら合計金額を印字します．

入力されたデータは，（数量×単価）で金額を計算し，区分コードごとに集計します．

合計金額の印字を，区分コード，合計金額の順に印字して終了するプログラムを作ってください．

なお，区分コード別の集計には配列を使ってください．

区分コード：1～9と99 数　量 単　価	⇨	99で合計印字 数量×単価を区分コード別に集計	⇨	区分コードと合計金額 CODE 1：¥9999

答えは 227 ページ⇨

9 プログラムコマンド⑸ その他の命令

1 STOP
プログラム実行の一時停止

命令	STOP	読み方：ストップ

この命令は，プログラムの実行を一時中断したい場合にプログラムの中に，書く命令です．

この命令の用途は，プログラムのデバッグやカセットテープにデータを記録する場合の待ちなどに効果的です．

1.1 STOPを使ったデバッグの手順

① プログラムを実行単位ごとに区切ってSTOPをそう入

⇩

② 実行が中断したらマニュアルモードで変数の内容を表示

⇩

③ 誤りの発生したプログラムのブロック（区切り）をさがす

⇩

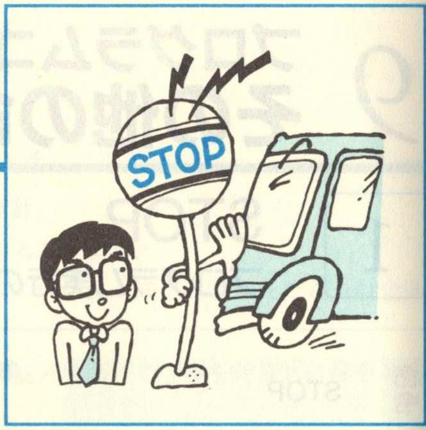
④ ③のプログラムブロックを細かく区切ってSTOPをそう入

⇩

⑤ 実行の中断ごとに変数の値をマニュアルモードで表示し，誤りの発生する行をさがす

⇩

⑥ プログラムの誤りを修正
STOPを削除する



たとえば，データを入力→計算→表示する，という3つのブロックに分けられるプログラムを仮定します．

そして，このプログラムが思う通りの結果が得られなかったとします．

このような場合，STOP命令を使い，前述の手順でデバッグをします．

まず，間違いがどの段階で発生したかを確認するのに，データ入力部の実行をSTOP命令で一時中断します．そして，変数をマニュアルで表示させて確かめます．

次に，計算（データの加工）が終わったところで，同じようにSTOP命令を使い，実行を一時中断し，変数を表示して確かめます．

表示，印字のブロックでも同様に確かめます．

大きなブロックでつかめたら，次第に範囲を狭めていき，バグを発見します．

1.2　STOPで止まったプログラムの再開

STOPを実行して，プログラムが中断した状態では，マニュアルで操作できます．

変数の表示はもちろん，数式の計算もできます．

たとえば，変数Aの内容を表示させるには……

A　EXE

……でしたね．

次に，途中で止まっていたプログラムを再開したいときは，単にEXEキーだけを入力してください．プログラムはSTOPした次から始まります．

もし，特殊キーを押して，WRTモードに切り換えてしまったら，EXEを押しても再開されません．

そのようなときは，1回確認が終っているプログラムブロックは再びチェックする必要はありませんから，RUN行番号でSTOPの書かれている次の行から実行を始めます．

2　END
プログラムの終了

命令	END	読み方：エンド

ENDは，プログラムがここで終ったことを，コンピュータにわからせる命令です．

前項のSTOPは一時中断ですから，EXEキーを押せばプログラムの実行を再開しました．しかし，ENDは実行終了ですから，EXEキーを押しても，何の反応もありません．

ここがSTOPとENDの違うところです．

プログラムにはEND命令がなくてもかまいません．後に続く行がなければそこをプログラムの終りであると，PB-300は判断してくれます．

しかし，END命令がないと困ることもあります．次の2つのプログラム例を比較してみてください．

☆　☆　☆

プログラム 15　センチメートルとインチ換算

このプログラムは，長さを入力すると，その値をセンチメートルと見なしてインチへ換算した値と，インチと見なしてセンチメートルへ換算した値の2つを表示します．

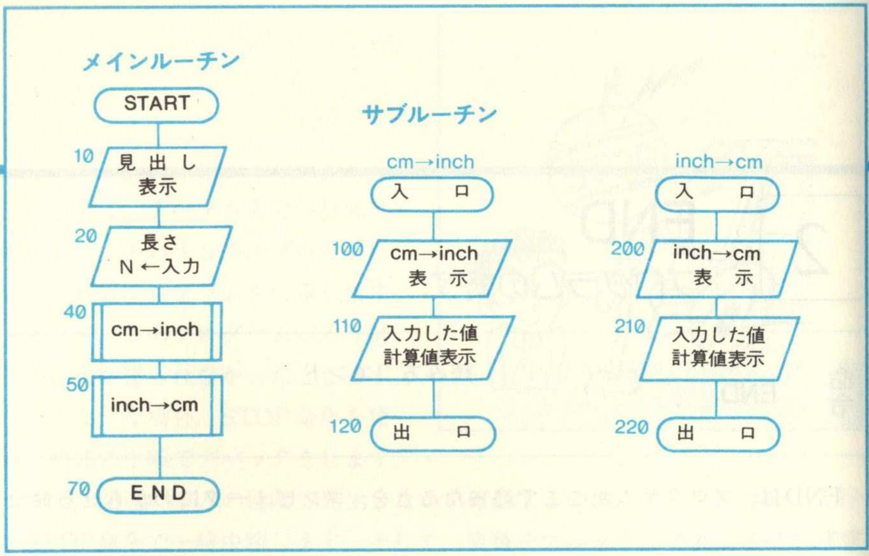
プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

■フローチャート



―――プログラムリストⒶ―――

```
10  PRINT "cm←→inch"
20  INPUT "Nagasa:", N
30  SET F2
40  GOSUB 100
50  GOSUB 200
60  SET N
100 PRINT "cm→inch"
110 PRINT N;"cm=";
    N/2.54;"inch"
120 RETURN
200 PRINT "inch→cm"
210 PRINT N;"inch=";
    N*2.54;"cm"
220 RETURN
```

―――プログラムリストⒷ―――

```
10  PRINT "cm←→inch"
20  INPUT "Nagasa:", N
30  SET F2
40  GOSUB 100
50  GOSUB 200
60  SET N
70  END
100 PRINT "cm→inch"
110 PRINT N;"cm=";
    N/2.54;"inch"
120 RETURN
200 PRINT "inch→cm"
210 PRINT N;"inch=";
    N*2.54;"cm"
220 RETURN
```

ない ← 70 END → ある

これだけが違う

プログラムリストⒶの実行時の表示

cm←→inch ⇨ EXE

Nagasa:? ⇨ 10 EXE

cm→inch ⇨ 以下EXEキーを押す

スクロール 10.00 cm= 3.94 inch

inch→cm

スクロール 10.00 inch= 25.40 cm

この部分がリストⒷと違う

cm→inch

スクロール 10 cm= 3.937007874 inch

ERR7 P0-120

プログラムリストⒷの実行時の表示

cm←→inch

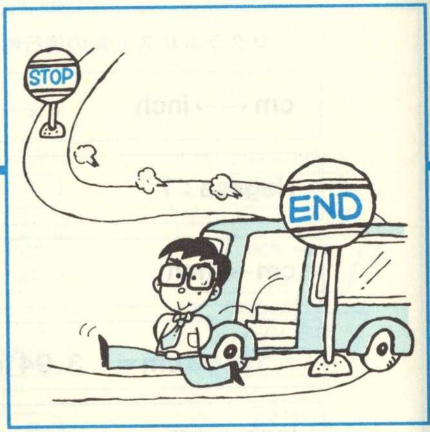
Nagasa:?

cm→inch

スクロール 10.00 cm= 3.94 inch

inch→cm

スクロール 10.00 inch= 25.40 cm



プログラム解説

リストⒶとリストⒷの違いは，70行に END 命令があるか，ないかの違いです．しかし，このちょっとした違いが，大変大きな違いになります．

両方のプログラムとも，メインルーチンと2つの表示サブルーチンで構成しています．

実行手順は，前ページのとおり，メインルーチンで長さNを入力したら，インチ換算のサブルーチンで表示計算し，メインルーチンへ戻り，今度は，センチメートル換算のサブルーチンで表示計算し，メインルーチンへ戻ります．

そして，表示桁数の指定を解除して，プログラムは終了するのですが……

リストⒶでは，END がないために，すぐ隣の 100 行を実行します．そして110行を実行し，表示します．表示桁数を指定していないため，ズラズラと長い表示となっていますね．

次に，120行はRETURNですが，今回はGOSUBで，この行まで来たわけではないため，戻れといわれても，戻るところがありません．

結局，ERR 7　P0-120 を表示して，プログラムは中断します．

この点，リストⒷは，70行にENDが書かれているため，サブルーチンの行へズカズカ踏み込むことはなく，ここで終了します．

どうやら，仕事は余計にやればよい，というものでもないようです．

そこで教訓を1つ……プログラムの終わりには「END」を書く習慣をつけよう……ENDを書くのがくせになるぐらいで，ちょうどいいのです．

3 VAC
変数をクリア

VAC命令は，マルチステートメントでは書けません．

命令	VAC	読み方：ブイ・エー・シー 語源　：Variable All Clear

PB－300の変数エリア26個と，DEFM命令で拡張した変数エリアを，すべて初期状態にする命令です．

変数の初期状態とは，数値変数では0に，文字変数ではNull（ヌル）のことをいいます．

VAC命令を実行すると，プログラムエリアに関係なく，すべての変数が初期状態に設定されます．

したがって，他のプログラムエリアに異なったプログラムが記憶されていて，変数に大切なデータが記憶されているような場合，このVAC命令を実行すると大変です．すべてが“水の泡”となりかねません．

他のプログラムとの関係に十分注意して使ってください．

ところで，VACと同様の機能を，代入文で実現することもできます．変数をすべてクリアしたのでは都合の悪い場合，次のようにします．

	数値変数	文字変数
変数	10 A=0:B=0:C=0	10 A$="":B$=""
配列	10 FOR I=1 TO 10 20 Z(I)=0 30 NEXT I	10 FOR I=1 TO 10 20 Z$(I)="" 30 NEXT I

このように，変数や文字変数は，必要な数だけ代入式を書きます．

配列の場合は，代入式とループを組み合わせて，初期値の設定を行ないます．

代入式による初期値設定の場合には，設定の必要な変数だけとし，マルチステートメントを使い，省メモリーを工夫します．

なお，プログラム中で使われるすべての変数を初期値設定する必要はありません．初期値設定の必要な変数や配列は，A＝A＋1　のように，元の値を計算して，また戻すという形で使う変数と，データを新しく記憶しようとする配列です．

☆　☆　☆

プログラム 16　0〜100点のデータを10クラスに分けてカウント．

このプログラムは，0から100点のデータを，0〜10，11〜20，………91〜100の10クラスに分け，それぞれのクラスごとの個数をカウントし，表示します．

入力できる数は，データの0〜100と，入力終了の合図としての−999です．−999を入力すると，各クラスのカウント数を順に表示します．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

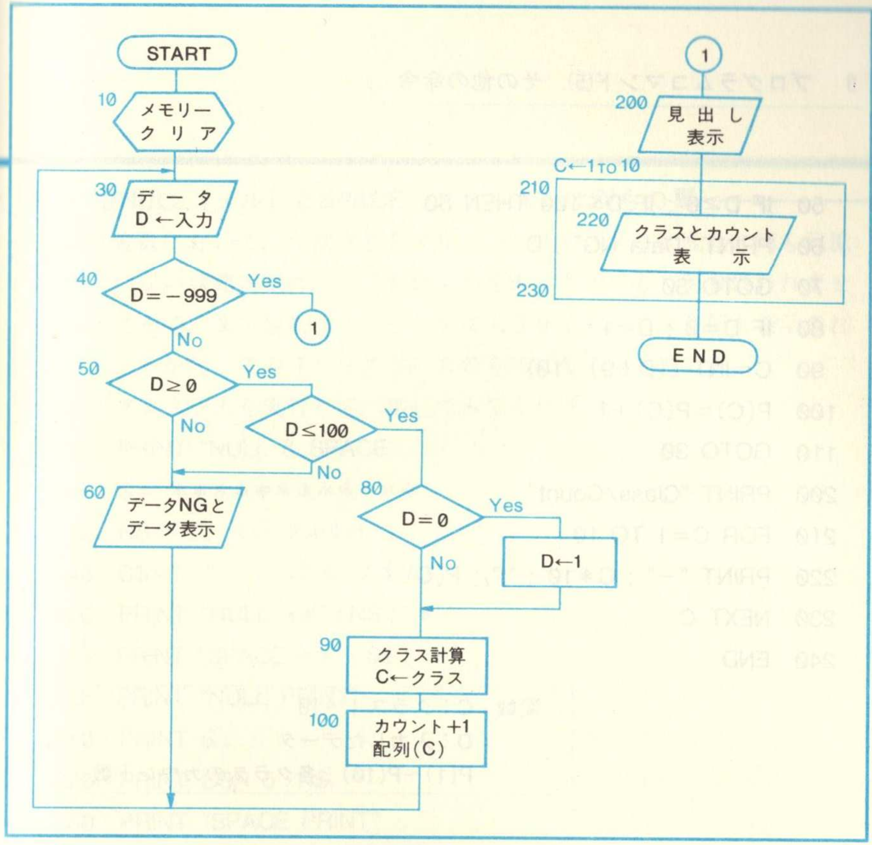
プログラム実行

MODE 0
RUN EXE

実行時の表示

表示	キー入力
＊ DOSUU ＊	⇨ EXE
Data：?	⇨ 35 EXE（正常なデータ）
Data：?	⇨ 604 EXE（異常なデータ） ⇩
Data NG 604	⇦ EXE
Data：?	⇨ −999 EXE（データ終了）

■フローチャート



表示	キー入力
Class／Count	⇨ EXE
− 10：5	⇨ EXE 各クラスとデータ表示
⋮	
− 100：0	⇨ EXE プログラム終了

プログラムリスト

```
10  VAC
20  PRINT "＊ DOSUU ＊"
30  INPUT "Data：", D
40  IF D=−999 THEN 200
```

```
 50  IF D≧0 ; IF D≦100 THEN 80
 60  PRINT "Data NG" ; D
 70  GOTO 30
 80  IF D=0 ; D=1
 90  C=INT ((D+9) /10)
100  P(C)=P(C)+1
110  GOTO 30
200  PRINT "Class/Count"
210  FOR C=1 TO 10
220  PRINT "-" ; C*10 ; ":" ; P(C)
230  NEXT C
240  END
```

変数　C：クラス１～10
　　　D：入力したデータ
　　　P(1)～P(10)：各クラスのカウント数

プログラム解説

配列に各カウント数を足していきます．

そのために，まず変数をクリアします．

データを入力すると，３つの条件文，４つの比較式を通過してクラス分けをします．

クラス分けの計算式は，INT((D+９)/10)　です．10の位への切り上げを行なった後に，10で割り整数化すると，１～10のクラスが得られます．

このために，80行で，データが０なら１に変更しています．

−999を入力したら，データ入力の終わりです．

200行へ分岐し，ループ命令で各クラスとカウント数を表示して，プログラムを終わります．

■NULL（ヌル）とSPACE（スペース）は別もの■

文字変数において，初期状態をヌルといいます．ヌルは変数に何も記憶されていない状態のため，この変数１つを表示させても，何も表われません．ところで，文字変数にスペースを代入させても，やはり何も表示されません．しかし，このヌルとスペースの意味は全く違うのです．

次のプログラムを実行して，試してみてください．

```
 10  PRINT "NULL & SPACE"
 20  $="************"
 30  N$=" " ………ヌルを代入
 40  S$="␣" ………スペースを代入
100  PRINT "NULL→*"; N$; "*"
110  PRINT "SPACE→*"; S$; "*"
220  PRINT "NULL PRINT"
210  PRINT $;
220  PRINT CSR 5; N$
250  PRINT "SPACE PRINT"
260  PRINT $;
270  PRINT CSR 5; S$
300  END
```

10行　NULL & SPACE

100行　NULL→**

110行　SPACE→*␣*

200行　NULL PRINT

210・220行　************

250行　SPACE PRINT

260・270行　*****　******

初めに，＊と＊の間にヌルとスペースをはさんで，それぞれ表示します．

次に，＊が12個表示されている６個目に，ヌルとスペースを表示します．

その結果は，スペースは１文字分空白ができ，ヌルは何も入りません．

4 DEG, RAD, GRA

角度の単位は３種類

書　式	表 示	単　位
MODE 4	DEG	度（デグリー）(degree)
MODE 5	RAD	ラジアン (radian)
MODE 6	GRA	グラジアン (gradient)

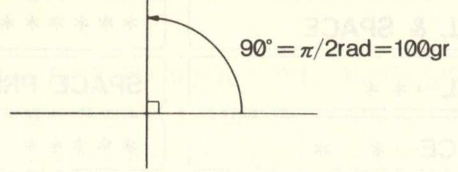
4，5，6は数値変数，式での指定も可能

三角関数，逆三角関数の計算を行なうには，まず角度の単位を指定します．

電源スイッチをONにした直後は，デグリー（MODE 4）に設定されます．

たとえば，sin xを計算するには，xが度で与えられていたら，MODE4(DEG)，ラジアンならば，MODE 5（RAD)，グラジアンなら，MODE 6（GRA)をプログラムの中に書きます．

逆三角関数，たとえば，$\cos^{-1}x$を計算するには，計算結果を度で表わしたいか，ラジアンかグラジアンかによって，MODE 4～6 を指定します．



$1° = \pi/180\text{rad} = 100/90\text{gr}$
$1\text{rad} = 180°/\pi$
$1\text{gr} = 90°/100$

プログラム　17　三角関数を希望する単位と範囲で計算

三角関数を計算するのに，どの関数の計算か，角度の単位は何か，角度はいくつからいくつまでを，いくら置きに計算するかの５項目を指定して計算します．

このプログラムは，三角関数の三冠王になれるかな？

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

Sin, Cos, Tan　⇨ EXE

1=Sin, 2=Cos, 3=Tan?　1を入力，関数を指定

4=DEG, 5=RAD, 6=GRA?　4を入力，角度単位指定

Kaku START=?　0を入力，角度の始まり

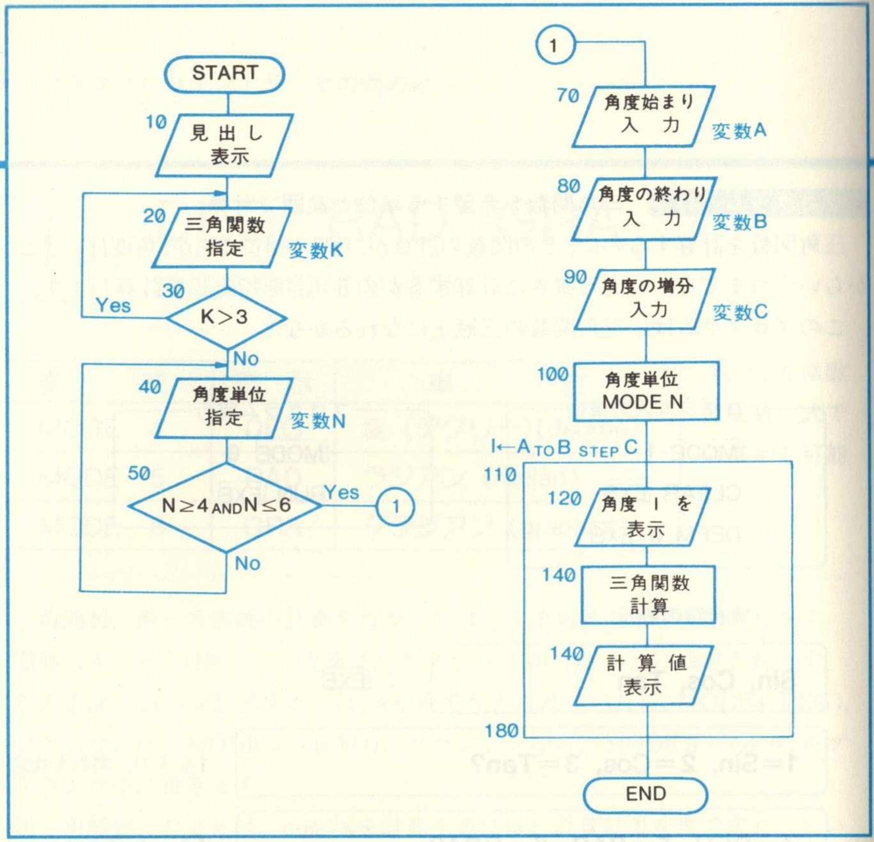
Kaku END=?　180を入力，角度の終わり

Kaku STEP=?　15を入力，角度の増分

計算結果⇨

```
  0 0.000        105 0.966
 15 0.259        120 0.866
 30 0.500        135 0.707
 45 0.707        150 0.500
 60 0.866        165 0.259
 75 0.966        180 0.000
 90 1.000
```

■フローチャート



プログラムリスト

```
10  PRINT "Sin, Cos, Tan"
20  INPUT "1=Sin, 2=Cos, 3=Tan", K
30  IF K>3 THEN 20
40  INPUT "4=DEG, 5=RAD, 6=GRA", N
50  IF N≧4 ; IF N≦6 THEN 70
60  GOTO 40
70  INPUT "Kaku START=", A
80  INPUT "Kaku END=", B
90  INPUT "Kaku STEP=", C
100 MODE N
110 FOR I=A TO B STEP C
```



```
120 PRINT I ;
130 SET F3
140 IF K=1 ; PRINT SIN I
150 IF K=2 ; PRINT COS I
160 IF K=3 ; PRINT TAN I
170 SET N
180 NEXT I
190 END
```

変数 A：角度の始まり
B：角度の終わり
C：角度の増分
I：角度の表示，計算（ループ）
K：三角関数の指定
N：角度単位指定

プログラムの解説

ループ命令と条件処理を組み合わせて，計算表示を行ないます．

入力は一定の条件に入らない場合は，再入力を要求します．角度の始まり，終わり，増分は入力のチェックはしません．

角度単位の指定は，変数Nで行ないます．変数で指定できるのは便利です．

10：見出しを表示する．

20：Sin，Cos，Tanのいずれで計算するかを，1，2，3で指定する．入力した値はKに入る．

30：Kが3より大きい場合は，規定外の数なので，再入力を要求するために20行へ分岐する．

40：角度の単位を指定する．入力する数はMODE nと合わせて，デグリーを4，ラジアンを5，グラジアンを6とする．入力した値はNに入る．

50：Nが4，5，6のいずれかであれば，プログラムを次に進める．そうで

なければ，60行を実行して，再度角度単位の指定へ戻る．

70：角度の始まりを入力．入力した値はAに入る．

80：角度の終わりを入力．入力した値はBに入る．

90：角度の増分を入力．入力した値はCに入る．

100：角度の単位を MODE Nで指定する．

110：ループ命令の始まり．

120：角度を表示．続けて計算結果を表示するので最後はセミコロン（；）．

130：三角関数の計算結果は，下3桁までの表示を指定．

140～160：Kの値により関数を選択．

170：表示桁数の指定を解除．

180：ループ命令の折り返し．

190：END命令．プログラムの終わり．

■省メモリーにマルチステートメント■

命令と命令，式などの間をコロン(：)で区切れば，62文字以内で1行にいくつもの命令，式を書くことができます．

このような書式をマルチステートメントと呼びます．

しかし，いくつも書けるからといって，何でもかんでも1行にまとめるのは感心しません．ちょっとした使い方のテクニックがあります．

このテクニックとは，サブルーチンのまとめ方と共通しています．サブルーチンは，1つの処理のまとまりごとに書きましたが，これと同じように，マルチステートメントも，一連の処理で命令の少ないものに使うと便利です．

そのプログラム例を，次に示しておきます．

```
100  INPUT "Data：", D：D=INT (D)：IF D<0
     THEN 100
100  GOSUB 500：GOTO 10
100  B=B*4-C：IF B<25 THEN 200
100  IF Z$="END"；GOSUB 1000：PRINT "Tokuten"
     ；A：END
100  FOR I=1 TO N：PRINT "No."；N；V(I)：NEXT I
```

1行プログラム

```
 10  PRINT CSR 11；"←"；：FOR I=1 TO 12：PRINT
     "␣"；：NEXT I：GOTO 10
```

10 組み込み関数を使う

PB−300の組み込み関数には，数値関数と文字関数の２種類があります．

巻末の命令一覧表276ページを見てください．

数値関数は，なじみのある記号が並んでいます．おそらく，使い方も予想がつくのではないでしょうか．

一方，文字関数を見ていただきますと，こちらは初めて見る記号ばかりだと思います．すでに，前章までのプログラム例で，これらのいくつかの関数を使っていますが，この章で，すべての関数についてまとめておきます．

1 数値関数

1.1 三角関数と逆三角関数

三角関数と逆三角関数

	一　般	PB−300	読み方
三角関数	sin	SIN	サイン
	cos	COS	コサイン
	tan	TAN	タンジェント
逆三角関数	$\sin^{-1}$	ASN	アークサイン
	$\cos^{-1}$	ACS	アークコサイン
	$\tan^{-1}$	ATN	アークタンジェント

三角関数は，角度の大きさから辺の比を求めるときに使います．逆三角関数は文字通りその逆で，辺の比から角度の大きさを求めます．

角度には単位があるので，三角関数と逆三角関数を使うときは，まず角度の単位の指定から行ないます．

角度の単位	指　定	表　示
デグリー	MODE 4	DEG
ラジアン	MODE 5	RAD
グラジアン	MODE 6	GRA

角度の単位については，170ページも合わせてご覧ください．

1.2 対数関数……LOG（ログ）

	一　般	PB−300	読み方
常用対数	$\log_{10}$	LOG	ログ
自然対数	$\log_e$	LN	ログ・ナチュラル

対数は，LOGあるいはLNに続いて，正の数値定数，変数，式を書きます．

```
LOG 233
LN X
```
……正しい使い方

```
LOG −500
```
⇨ 負のため……ERR3

上記のように，負の値をとると，ERR 3が発生します．

プログラムの中で使用していて，変数が負の値をとる可能性のある場合，次のような対策をとるとよいでしょう．

① 負の数なら計算しなくてよい場合

```
100 IF X<0 THEN 120  (あるいは目的の行番号)
110 L=LOG X
120 ………
```

② 負の数でも計算はするが，結果には正負の区別を必要としない場合

```
100 L=LOG (ABS X)  ……Xの絶対値で計算する
```

③ 負の数でも計算し，結果に正負の区別が必要な場合

```
100 L=LOG (ABS X)*SGN X  ……Xの正負の符号を掛ける
```

1.3 指数関数……EXP（エキスポーネント）

	一　般	PB−300
自然対数の底eのべき乗	e	EXP 1
xのy乗	x^y	X↑Y

eは自然対数の底で，値は……

2.718281828

……です．

EXP 1 EXEによって，PB－300はこの値を表示してくれます．



xのy乗は，xをy回掛けるという意味ですが，よくご存知だと思います．この場合，xは0より大きいことが条件です．

EXPのような関数の次にくる数値のことを引数と呼びます．

引数として使える数値は，関数ごとに異なりますが，EXPの場合は，1だけです．

また，引数として書けるのは，数値であれば，定数，変数，式のいずれでも可能です．

定数や変数は書き方での問題はないのですが，式では注意してもらいたいことが1つあります．

次の2つの例を試してください．

ここでは，xのN×2乗を計算したいのですが……

① X↑N*2

② X↑（N*2）

……①②では結果が違います．①②の計算の意味は，それぞれ次のようになります．

① XのN乗を計算して，その結果を2倍する．

② 目的通り，XのN×2乗を計算する．

つまり，演算記号には必ず優先順があります．

（　），関数，↑，×と÷，＋と－，＝の順で，前の方が優先されます．

これで，①と②が違ってくることがおわかりになるでしょう．

指数関数に限らず，数式の書き方には十分注意してください．関数で計算させたい範囲は，（　）で囲むことをお忘れないように．

1.4 平方根……SQR（スクウエアールート）

一　般	PB－300
$\sqrt{x}$	SQR X
$\sqrt[3]{x}$	X↑(1/3)

平方根は$x^{\frac{1}{2}}$のことです．1/2乗は$\sqrt{\ }$で表わしますが，PB－300ではSQRと書きます．

3乗根は$x^{\frac{1}{3}}$のことですから，SQRは使うことはできず，X↑(1/3)と書きます．この例にならえば，平方根はX↑（1/2）と書くこともできます．

SQRの引数は，数値定数，変数，式で書くことができますが，引数が負になると，ERR 3が発生します．

プログラムの中で使っていると，変数（引数）が負の値をとる可能性があります．

その場合の対策例を次に示します．これらは対数関数の対策法と同じです．

① 負の数なら計算しなくてよい場合

```
100 IF X<0 THEN 120 （あるいは目的の行番号）
110 S=SQR X
120 ………
```

② 負の数でも計算はするが，結果には正，負の区別を必要としない場合

```
100 S=SQR (ABS X)
```

③ 負の数でも計算し，結果に正負の区別が必要な場合

```
100 S=SQR (ABS X)*SGN X
```

式の平方根を計算するときは，（　）を付けることを忘れないように．

1.5 符号化……SGN（サイン）

Xの値	SGN Xの値
0より小さいとき	−1
0のとき	0
0より大きいとき	1

SGN関数は，引数の符号化に利用します．

これと同じ働きを条件文で書くと次のようになります．

```
100 IF X<0;S=-1
110 IF X=0;S=0
120 IF X>0;S=1
```

このような3つの働きを，1つの関数でこなすことができるわけですから便利ですね．

この関数の用途は，対数関数と平方根でプログラム例で使ったように，正負の符号を付けるとか，処理の中で値によって3つの方法を選択するスイッチの役目です．

なお，SGNの引数には，数値定数，変数，式を書きます．式は(　)で囲ってください．

1.6 乱数……RAN＃（ランダム）

RAN＃と書くことによって，0＜RAN＃＜1の小数第9位までの乱数を得ることができます．

乱数は，他の変数に代入して……

```
R=RAN#
```

……のように使うこともありますし，……

```
PRINT (INT (RAN#*10))
```

……のように，数式や関数に組み込んで使うこともあります．

用途は，ゲームや無作為抽出のための乱数表，シミュレーション用など，幅広く使うことができます．

RAN＃関数は，小数表示なので，発生した乱数をいろいろと加工して使います．その事例を次に示します．

① 希望する桁までの整数を取り出す

```
INT (RAN#*10↑L)
```
……Lは桁数

② 0から上限Mまでの整数を取り出す

```
RND (RAN#*M, -1)
```

③ 1から上限Mまでの整数を取り出す

```
INT (RAN#*M)+1
```

1.7 四捨五入……RND（ラウンド）

RND（X，Y）によって，Xの値を10^Yの桁で四捨五入します．

たとえば，四捨五入した値を変数Rに代入するときは……

```
R=RND (X, Y)
```

これと類似した関数にINTとSET Fがあります．

INTは，小数点以下を切り捨てる関数で，演算式に使うことができます．

SET Fは，指定した桁数の1つ下の桁を四捨五入しますが，演算式には使えず，表示する桁数を制御するコマンドです．

RND関数の引数XとYは，数式，変数，式が書けます．なお，Yの値が小数の場合は，小数以下を切り捨てます．

たとえば，RND（1.2345，−4）の結果は1.235となります．

1.8 絶対値化……ABS（アブソリュート）

数学で使う，|x|（絶対値）をPB−300では，ABS Xと書き，Xの絶対値を得ることができます．

X＊SGN X のように，SGN関数でも絶対値を得ることができますが，このように書くよりも1つの関数で書くことができるので便利です．



用途は，対数関数や平方根に出てきた例のように，負のままで計算したくない場合に使われます．

引数には，数値定数，変数，式を書きます．

また，式を書く場合には，（　）で囲ってください．

1.9　整数化……INT（インテジャー）

INT X と書くことによって，Xの値の小数部を切り捨てます．

すでに，これまでのプログラム例の中でしばしば出てきましたので，およそおわかりだと思います．

正の数値Xを，たとえば，整数部と小数部に分けるとすれば，INT関数とFRAC関数を使って，次のようにします．

A＝INT X：B＝FRAC X　……Aに整数部，Bに小数部が代入される

引数には，数値定数，変数，式が使えます．

式は（　）で囲ってください．

INT関数を使うときに，注意が必要な点は……

① 正の数なら………そのまま整数部のみ

② 負の数なら………｜X｜の整数部＋1の値に－符号を付けた値

……のように，正負で異なることです．

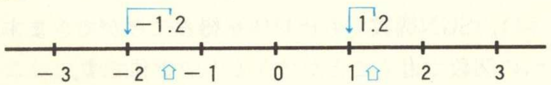
たとえば，………

1.2 ……… INT1.2＝1

－1.2 ……… INT－1.2＝－2

……となります．

これは，INT Xは，Xを越えない最大の整数を与えるという性質によるもので，間違いではありません．



1.10　整数部除去……FRAC（フラクション）

FRAC X と書くことにより，引数Xの小数部のみを取り出すことができます．

たとえば……

FRAC 1.2＝0.2

FRAC －1.2＝－0.2

……となります．

このように，小数部だけを取り出す関数ですから，すなおに整数部を除去した値となります．

引数には，数値定数，変数，式を書きます．

式は（　）で囲ってください．

2　文字関数

文字関数には，次の3種類が用意されています．

LEN	文字数を求める	文字変数，$
VAL	文字列の数字を数値に変える	文字変数，$
MID	文字を指定した数だけ取り出す	専用文字変数($)

LENとVAL関数は，文字変数と専用文字変数に使えますが，MID 関数は専用文字変数（$）にしか使えません．

文字関数の次にくる文字変数，または専用文字変数（$）は，必ず（　）で囲ってください．そうしないと，ERR 2が発生します．

2.1　文字数を求める……LEN（レングス）

文字変数や専用文字変数に記憶されている文字数が何文字かを求めます．

L=LEN（$）

用途は，文字数を制限して入力するときや，文字を右づめ，左づめで表示，印字する場合などです．（これらのプログラム例は，217ページを参照）

文字定数を，そのままLEN関数に使うとERR2が出ます．このようなときは，文字列を一旦，文字変数（専用文字変数がよい）に代入してから，文字数を求めてください．

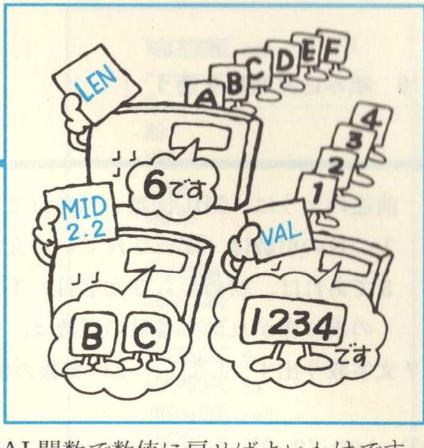
ヌル状態の文字変数の長さは0ですが，スペースは数に数えられます．

2.2　文字列の数字を数値に変換……VAL（バリュー）

VAL関数は，文字変数に記憶されている数字を数値に変換します．

V=VAL（A$）

PB－300で扱える数字には2種類あります．1つは文字としての数字と，もう1つは数値としての数字です．つまり，文字としての数字とは文字変数に記



憶されている数字であり，数値としての数字とは，数値変数に記憶されている数字のことです．

文字変数へ数字を記憶させておけば，表示を一番左から（符号がない）ということも可能ですが，この数字を使って計算させる必要がある場合，VAL関数で数値に戻せばよいわけです．

```
100　$=KEY：IF $=" " THEN 100
110　IF $≧"1"；IF $≦"5"；GOTO VAL（$）*1000
120　GOTO 100
```

KEY命令の代入式の左辺は文字変数と決まっています．押されたキーが数字のキーでも，$に代入されるのは文字としての数字です．

この数字をそのまま，GOTO命令の分岐先の行番号の数値としたいために，VAL関数を使っています．

つまり，1～5までのキーが押されたら，1000～5000行へ分岐します．

このようにしておくと，もし誤って，数字キーではなく他の文字キーを押しても，100行に戻ることができます．

もう1つ例をあげておきます．

```
100　A$="400"
110　L=N+A$　……ERR2が発生
110　L=N+VAL（A$）……計算できる
```

2.3　文字変数から取り出す……MID（ミッド）

専用文字変数$に記憶されている文字列の一部を取り出すときに使います．

```
PRINT MID（3, 7）
A$=MID（1, L）
```

前述のように，MID関数の引数は2つあります．

1つ目の引数は，記憶されている文字の初めから何文字目かを表わします．3であれば，初めから3文字目，1であれば，最初の文字ということです．

，の次に書かれた2番目の引数は，何文字取り出すかを表わします．7なら7文字取り出す．Lなら，その変数の値だけ取り出すことを意味します．

$="Dr . SLAMP"

① MID (3, 6) …… 3文字目から6文字………… . SLAMP
② MID (4, 5) …… 4文字目から5文字………… SLAMP
③ MID (3, 7) …… 3文字目から7文字………… ERR5
④ MID (9, 1) …… 9文字目から1文字………… ERR5
⑤ MID (1) ……… 1文字目からすべて………… Dr . SLAMP
⑥ MID (4) ……… 4文字目からすべて………… SLAMP

③④の例のように，1番目，2番目の引数が記憶されている文字数よりオーバーしたらERR 5が出ます．

また，⑤⑥の例のように，2番目の引数を省略することができ，そのときは1番目の引数で指定された文字から最後まで取り出します．

ただし，1番目の引数を省略して，MID（，4）のように書くことはできません．引数には，数値定数，変数，式を書くことができます．

☆ ☆ ☆

プログラム 18 よいしょプリント

LENとMID関数，およびループ命令を組み合わせて，入力した名前をプリントアウトするプログラム例です．

名前をたくさんプリントして，“エライ”ともち上げるので，“よいしょプリント”ということにしておきます．

プログラム入力準備

MODE 1
CLEAR EXE
DEFM 0 EXE

プログラム実行

MODE 0
RUN EXE

実行時の表示

MOJI KANSUU

⇨ EXE キーを押す

NAME : ?

⇨名前を入力

```
NAKASONE san wa ERAI
--------------------
N
NA
NAK
NAKA
NAKAS
NAKASO
NAKASON
NAKASONE
NAKASONE
NAKASONE s
NAKASONE sa
NAKASONE san
NAKASONE san
NAKASONE san w
NAKASONE san wa
NAKASONE san wa
NAKASONE san wa E
NAKASONE san wa ER
NAKASONE san wa ERA
NAKASONE san wa ERAI
```

プログラムリスト

```
10 PRINT "MOJI KANSUU"
20 INPUT "NAME : ", $
30 L=LEN ($) : IF L>8 ; L=8
40 $=MID (1, L)+" san wa ERAI"
50 L=LEN ($)
60 MODE 7
70 PRINT $, "--------------------"
80 FOR I=1 TO L
90 PRINT MID (1, I)
100 NEXT I
110 MODE 8
120 END
```

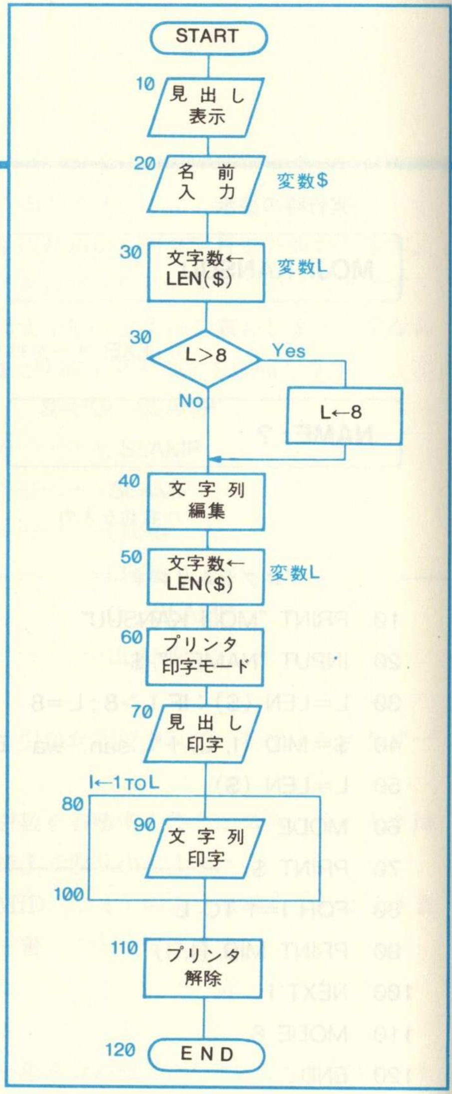
変数	
I	ループ
L	文字数
$	名前の入力と文字列

プログラム解説

入力した名前は$に記憶します．

$の文字数を上限8文字に設定して，“ san wa ERAI”を足します．これでプリンタに印字する文字列ができたので，1文字ずつ順に増やしながら印字します．印字する文字はMID（1，I）で，1文字目からループカウンタの値までとします．

■フローチャート



10：見出しを表示する．

20：名前を入力する．変数は$

30：入力した名前の文字数をLに代入する．Lが8文字以上なら，8文字を代入する（文字数の上限を8文字とする）．

40：入力した名前の1文字目からL文字と，文字定数を足して$に代入する．

50：文字数をLに代入する．

60：プリンタを印字モードに切り換える．この後，実行するPRINT命令は，プリンタへ出力される．

70：見出しを印字する．

80：ループ命令の始まり．

90：MID（1，I)で文字列を印字する．印字するのは1文字目からI文字．

100：ループ命令の折り返し．

110：プリンタモードの解除．

120：END命令，プログラムの終了．

〔練習問題5〕　キーワード

A～Zのうち1文字を乱数で発生させ，その文字をできるだけ速く見つける“KEY WORD”ゲームを作りましょう．

キーワードを見つけるために，いくつかの文字を入力し，その文字列にキーワードが含まれているときは，“GOOD”を一定時間表示し，含んでいなかったら，“NG”を一定時間表示して，再び文字列の入力に戻ります．

これを何度か繰り返し，最後に，キーワード1文字が入力されたら，“VERY GOOD→→KEY WORD”と表示して終わりとします．

なお，A～Zの1文字を乱数で発生させるサブルーチンは，次のプログラムを利用してください．

```
200  $="ABCD………XYZ"
210  I=INT (RAN#*26) +1
220  A$=MID (I, 1)
230  RETURN
```

また，キーワードを見つけるために，文字列を入力する文字変数には，もう一度，$を用いてください．そして，FOR・NEXTループとMID関数で，入力した文字列にキーワードが含まれているかどうかを判断させてください．

答えは228ページ⇨

11 カセットテープレコーダの利用

1 PB−300の能力を10倍上げる方法

PB−300の命令を一通り覚え，使い方もわかってくるにつれ，出来上がるプログラムの数も増えます．また，プログラム１本当りのステップ数も大きくなります．

PB−300には，プログラムエリアが10ヵ所あり（P０～P９），独立した10本のプログラムを記憶できますが，それ以上の本数，あるいはステップ数が足りなくなると，記憶させることはできません．

そこで威力を発揮するのが，カセットテープレコーダです．

つまり，出来上がったプログラムをテープへ記録して，保管することができるからです．

このようにしておけば，PB−300の残りステップ数をさほど気にすることもなくなります．

プログラムだけではなく，データも同様にカセットテープへ記録できます．

今日までの合計を明日また使う，といった場合，カセットテープへ記録しておけばいいわけです．

もちろん，一度カセットテープへ記録したプログラムやデータは，いつでもPB−300へ読み出すことができます．

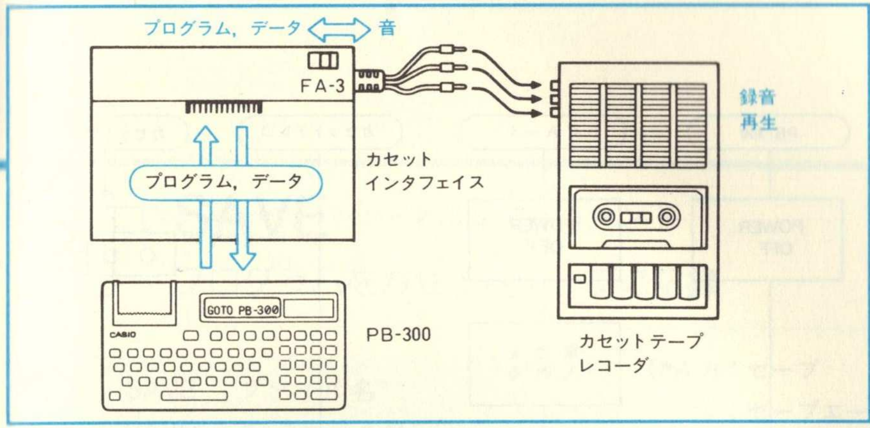
PB−300の内部に持つメモリーを主メモリーとか内部メモリーと呼ぶのに対し，カセットテープのように，コンピュータ本体の外側に記録する装置のことを，外部メモリーと呼んでいます．

外部メモリーを利用すれば，PB−300の能力を10倍も20倍も上げることができるわけです．

1.1 カセットテープレコーダを使うためのハードウェア

PB−300に記録されているプログラムやデータを，カセットテープへ記録するには，カセットインタフェイスFA−３が必要です．

■PB−300，FA−3とカセットテープレコーダの接続



FA−３は，PB−300のプログラムやデータを音(耳に聞こえる＝カセットテープに録音できる）に変換します．ただ変換するだけではなく，一定の決まったフォーマットへ変換しています．

カセットテープからPB−300へ読み出すときには，音をプログラムやデータに変換する働きをします．

このように変換する働きは，カセットインタフェイスがルールを決めて，手助けをしています．

1.2 PB−300，FA−3とカセットテープレコーダの接続

PB−300とFA−3の間のプログラムやデータのやり取りは，中央部にある12本のピンを通じて行ないます．

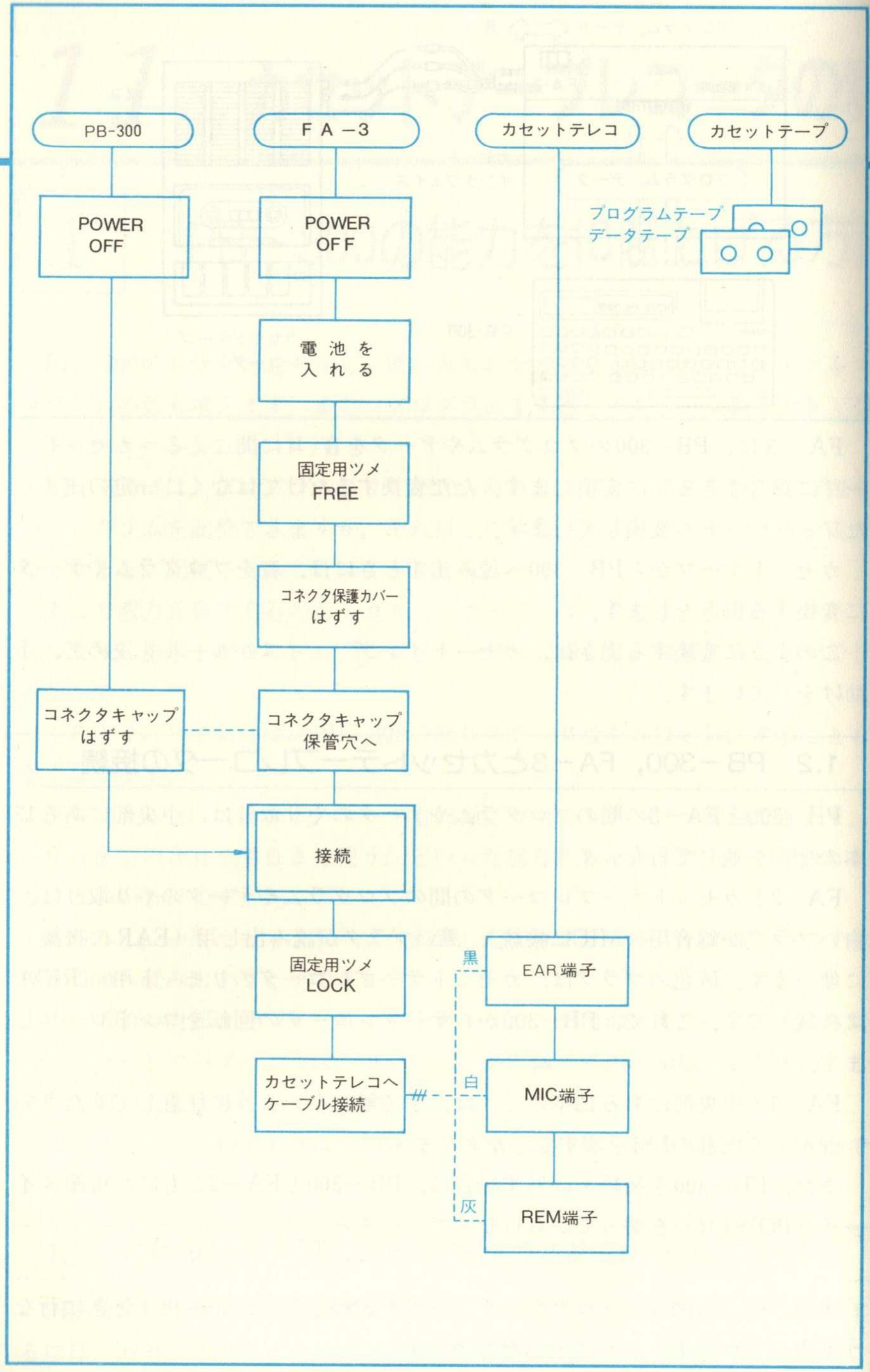
FA−3とカセットテープレコーダの間のプログラムやデータのやり取りは，白いプラグが録音用（MICに接続），黒いプラグが読み出し用（EARに接続）に使います．灰色のプラグは，カセットテープレコーダのリモート用（REMに接続）です．これで，PB−300からテープレコーダの回転をコントロールします．

FA−3の中央部にある12本のピンは，手で触れないように注意してください．

静電気で内部のLSIを壊すことがあります．

また，PB—300を接続又は外す場合は，PB—300とFA—3ともに，電源スイッチ（POWER）を切ってから行なってください．

次のページの図は，プログラムやデータを記録したり，読み出すときに行なう準備をフローチャートに示したものです．



2 SAVE ……マニュアルコマンド

プログラムをカセットテープに記録

命令	SAVE "ファイル名" SAVE A "ファイル名"	読み方：セーブ ：セーブエー
書式例	例1. SAVE "PRO−1" 例2. SAVE 例3. SAVE A "FILE" 例4. SAVE A	Aの語源：All
実行	MODE 0（RUNモード）いずれのモード MODE 1（WRTモード）でも実行可能	

2.1 ファイル名

ファイル名は，カセットテープへプログラムを記録するときの呼び名です．

8文字以内であれば，どんな文字，どんな記号（"を除く）を使ってもかまいません．

省略する（書かない）ことも可能ですが，テープからプログラムを読み出すときに，名前がないので不便です．日時を経て見なおしたときに，何のプログラムであったか，わからなくなってしまいます．必ずファイル名を付けてください．

ファイル名は，プログラムがどんなものかを知る手掛かりになるものですから，必ずしもプログラム名と同じにする必要はありません．プログラム名が連想できるような名前を付けます．



2.2 SAVE “ファイル名” ……例1，例2

PB−300で現在指定されているプログラムエリア（P0～P9のうち1つ）のプログラムを，カセットテープへ記録します．

〈例2〉では，ファイル名を省略していますが，同じように記録されます．

2.3 SAVE A “ファイル名” ……例3，例4

PB−300のすべてのプログラムエリア（P0～P9）に記憶されているプログラムをカセットへ記録します．

PB−300では，プログラムエリア間のGOTO#，GOSUB#による分岐が可能なので，1つのプログラムエリアに限ったプログラムが作られるとは限りません．いくつものプログラムエリアを有効に使ったプログラムを作成した場合には，SAVE Aでカセットテープへ記録します．

読み出すときは，自動的に各プログラムエリアへプログラムが戻ります．

〈例4〉では，ファイル名を省略していますが，同じように記録されます．

2.4 プログラムをカセットへ記録する手順

カセットインタフェイスFA−3の電源をONにします．

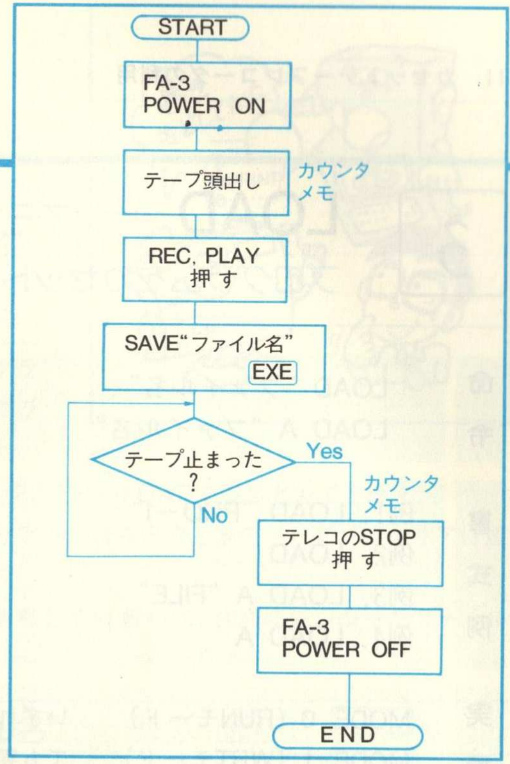
次に，テープを録音を始める位置まで進めます．テープにすでにプログラムが記録されているなら，間隔にも注意して，録音を始める位置を決めてください．このとき，カセットテープレコーダへリモート端子(灰色)が差し込んだままでは，テープが回らないので，テープの頭出しをする間だけ抜きます．

テープの位置が決まったら，リモート端子を差し込んで，テープカウンタの値をメモしておきます．

カセットテープレコーダを録音状態にセットします．

リモート端子が差し込まれているので，テープは回りません（早送り，巻戻し

■フローチャート



も不可），次に，PB−300へSAVE命令を入力します．

そしてEXEキーを押すと，カチッという音がして，テープが回り始めます．これでプログラムがカセットテープへ記録され始めました．

プログラムの記録が終わると，テープは止まります．

終了したら，テープカウンタの値をメモしてください．

カセットインデックスへファイル名，プログラム名，テープカウンタの初めと終わりの値を書いておきます．これで，プログラムの記録はすべて終わりです．

FA−3の電源をOFFにしてください．FA−3は，電源スイッチがONになっていれば，使わなくてもわずかな電流が流れます．

省エネのためにも，こまめに電源スイッチをOFFにしてください．

なお，リモートを使わなければ（リモート端子を差し込まない），FA−3のスイッチをONにする必要はありません．

2.5 プログラムの記録間隔

先ほど，テープにすでにプログラムが記録されているなら，間隔に注意してくださいと書きました．一通り出来上ったプログラムの後に記録する場合は，テープカウンタで10間隔をとれば十分です．ただし，入力途中の未完成のプログラムの後にはプログラムを記録しないでください．未完成のプログラムはこの後，どの程度ステップ数が増えるかわからないので，再び記録した場合，後のプログラムを消す恐れがあります．

3 LOAD ………マニュアルコマンド

プログラムをカセットから読む

命令	LOAD "ファイル名" LOAD A "ファイル名"	読み方：ロード ：ロードエー
書式例	例1. LOAD "PRO−1" 例2. LOAD 例3. LOAD A "FILE" 例4. LOAD A	
実行	MODE 0（RUNモード） MODE 1（WRTモード）	いずれのモードでも実行可能

3.1 ファイル名

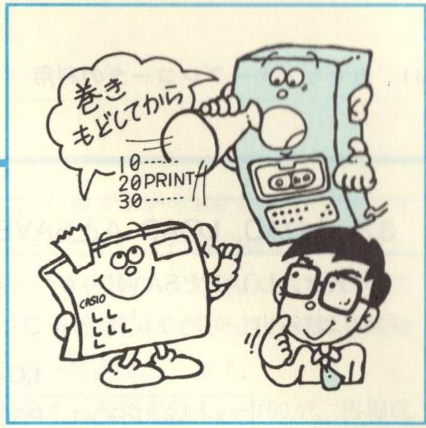
LOAD命令のファイル名の付けかた，あるいは省略はSAVE 命令の方法と全く同じです．

しかし，LOAD命令のファイル名の働きは，SAVE命令のそれと異なります．LOAD命令でのファイル名は，検索項目としての役目をもちます．

PB−300へ入力したファイル名と，テープに記録されているファイル名が一致するまで，プログラムを読み込まず，テープを回し続けて，さがします．

同じファイル名が見つかったら，そこでプログラムをテープからPB−300へ読み込みます．

ファイル名を省略した場合は，一番初めに見つかったファイル名のプログラムを読み込みます．



3.2 LOAD "ファイル名" ……例1，例2

SAVE命令で記録されたプログラムを，現在指定されているプログラムエリアへ読み込みます．

P6からSAVEしたプログラムを，P0へLOADすることもできます．

記録したときのプログラムエリアと，読み出すときのプログラムエリアは違っていてもかまいません．

〈例2〉では，ファイル名を省略していますが，初めに見つかったファイル名のプログラムをLOADします．

3.3 LOAD A "ファイル名" ……例3，例4

SAVE A命令でテープへ記録されているプログラムを読み出します．

プログラムエリアの指定はどこにあっても，記録したときのプログラムエリアへ，それぞれのプログラムは読み込まれます．

〈例4〉は，ファイル名を省略していますが，初めに見つかったファイル名のプログラムをLOADします．

3.4 ファイル名の表示

LOADまたはLOAD Aを実行して，テープに記録されているファイルを見つけると，PB−300は次のようにファイル名を表示します．

記録したときの命令	ファイル名	表示するファイル名
SAVE	PRO−1	PF：PRO−1
SAVE A	FILE	AF：FILE

ファイル名の表示は，こんなファイルを見つけました，というメッセージです．

3.5 LOAD, LOAD AとSAVE, SAVE Aの関係

次の表は，LOADとSAVEの4つの命令の関係を示したものです．

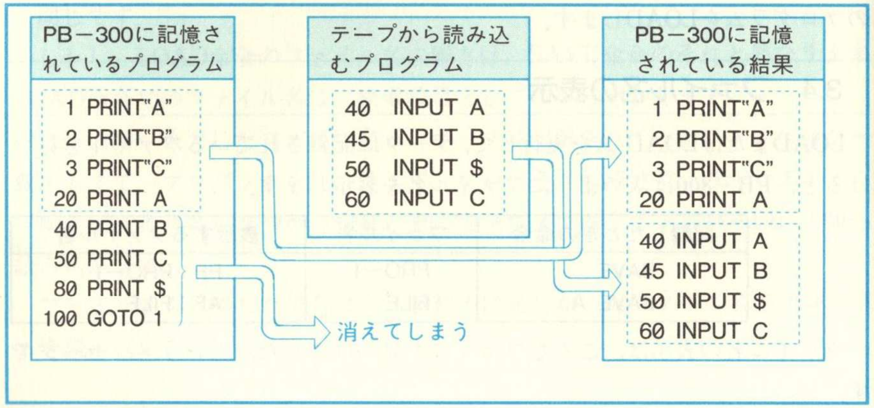
○印は読み出し可能な組み合せです．

		LOAD			LOAD A		
	ファイル名	PRO-1	PRO-2	なし	FILE	KASAN	なし
SAVE	PRO-1	○	×	○			
	PRO-2	×	○	○			
	なし	×	×	○			
SAVE A	FILE				○	×	○
	KASAN				×	○	○
	なし				×	×	○

3.6 プログラムの記憶されているエリアへ読み込んだら

すでにプログラムの記憶されているプログラムエリアへ，テープから別のプログラムを読み込んだらどうなるでしょうか．

その様子を示したのが，次の図です．



PB－300へ記憶されているプログラムは，行番号1から100です．

一方，テープからPB－300へ読み込むプログラムは，行番号40から60です．

これをPB－300へLOADしますと，最初にPB－300に記憶されていたプログラムは，テープから読み込むプログラムの最小行番号40より小さい行番号は残っていますが，それより大きい行番号は，すべて消えています．

このようにして，2つのプログラムが合わさった行番号1から60が，再編成されてPB－300に残ります．

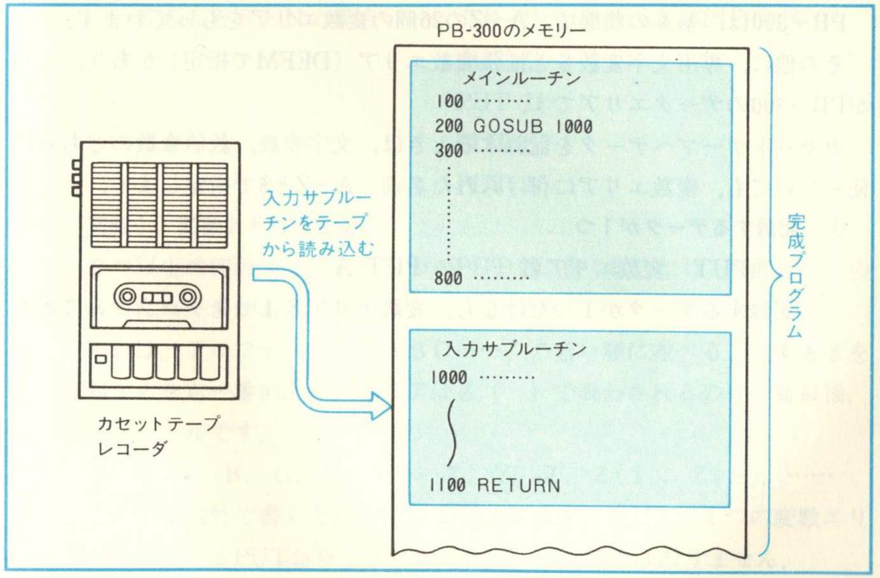
以上は，誤ってLOADしてしまい，プログラムを壊す結果になってしまいましたが，これを逆に利用する方法もあります．

たとえば，下の図のように，まず，1000番台を入力ルーチンとしてテープに記録しておきます．

次にメインルーチンのプログラムをPB－300に入力しておきます．

そして，テープから入力ルーチンのプログラムを読み込むと，2つのプログラムが合成され，1本のプログラムに完成させることができます．

便利な使い方ですから，利用されるとよいでしょう．



4 PUT ………プログラムコマンド
データをカセットテープに記録

命令	PUT“ファイル名”変数1，変数2,……	読み方：プット
書式例	例1．PUT “DATA” A 例2．PUT $, A, E 例3．PUT D, Z 例4．PUT $, A, Z (20)	
実行	行番号なしのマニュアルモードでも実行可能	

4.1 変数

PB－300は，基本の状態で，A～Zの26個の変数エリアをもっています．

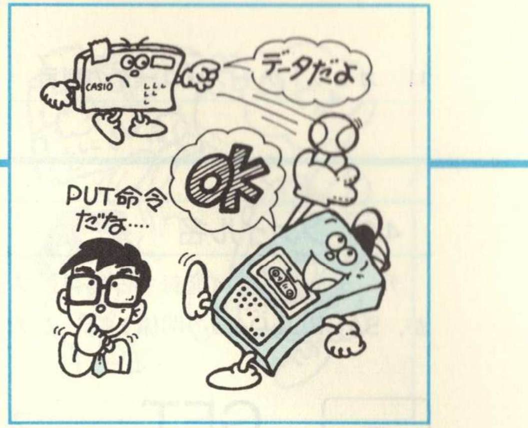
その他に，専用文字変数$と拡張変数エリア（DEFMで指定）があり，これがPB－300のデータエリアです．

カセットテープへデータを記録するときは，文字変数，数値変数のどちらに使っていても，変数エリアに付けられた名前，A～Zと$で指定します．

① 記録するデータが1つ

PUT 変数エリア名 ……… PUT A

記録するデータが1つだけなら，変数エリア名1つをプログラムに書きます．



② 記録するデータが2つ以上

■連続する変数エリア

PUT 変数1，変数2

……PUT A，E

AからEまでの変数エリアを記録したい場合は，変数エリアの初まり（A）と変数エリアの終わり（E）を書くことで，間にはさまれた変数エリアのデータも記録します．

PUT D，Z (20) ……では，DからZまでの変数エリアと，拡張エリアのZ(1)からZ(20)までに記憶されているデータが記録できます．

■連続しない変数エリア

$とAからZの場合 ⟶ PUT $，A，Z

$とAとCの場合 ⟶ 2行に分けて書く

PUT $，A

PUT C

GとLとSとNの場合 ⟶ それぞれの変数ごとにPUTを書く

PUT G

PUT L

PUT S

PUT N

③ 変数1，変数2を書く順番

2つ以上の変数エリア名を1つのPUT文に書く場合には，変数エリアの書き方にルールがあります．

それは，アルファベット（ABC……）の若い順に並べることと，$を記録するなら一番前，拡張エリアはZ（ ）で表わされるので一番最後，というルールです．

$，A，B，C，D，………X，Y，Z，Z(1)，Z(2)，……

この順番以外で書くと，ERR 2が発生します．ただし，1つの変数エリアだけ指定するPUT命令どうしであれば，この制限はありません．

4.2　ファイル名

データをテープに記録する場合，ファイル名を省略して書くこともできますが，SAVE，LOADの関係と同じように，付けた方が便利です．

5　GET ………プログラムコマンド
データをカセットテープから読む

命令	GET “ファイル名” 変数1，変数2，……
書式例	例1．GET “DATA” A 例2．GET $, A, E 例3．GET D, Z 例4．GET $, A, Z (20)
実行	行番号なしのマニュアルモードでも実行可能

読み方：ゲット

5.1　変数

変数エリアの名前の付け方はPUTと同じです．

PUTで記録したデータを読み出すには，PUT で記録した変数エリアと同数の変数エリアをGETで指定します．

普通は，PUTで使った書式を，そのままGETでも使います．

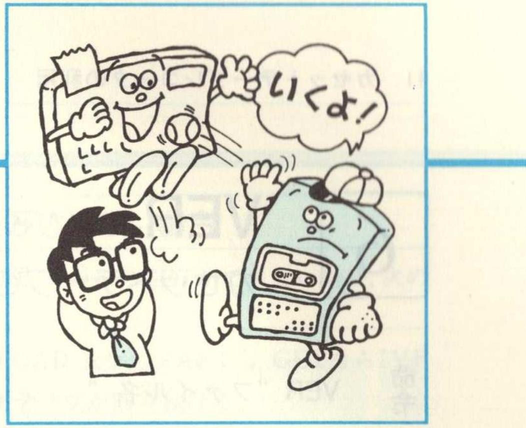
PUT $，A，E　⟶　GET $，A，E

PUT D，I　⟶　GET Z(1)，Z(6)

上記のいずれでも正常に読み込まれます．

① PUTの変数エリア数 ＞ GETの変数エリア数

PUT A，E　⟶　GET A，C



この場合，5件のデータがテープに記録されていますが，読み込む方は，3件分の変数エリアしか用意されていません．

このようなときは，データを先頭から3件を読み込んで，GET命令の実行を終了します．

② PUTの変数エリア数 ＜ GETの変数エリア数

PUT A，E　⟶　GET A，Z

5件のデータが記録されています．一方，読み込む方は26個分のデータを要求する命令です．

このままGET命令を実行すると，データを読み込み，データファイルが終わったところで，GET命令の実行を終了します．

変数エリアは，GET命令に書かれた初めの方から，データファイルに記録されている数だけ（例では5件）入力されます．

5.2　ファイル名

ファイル名は省略可能です．

ただし，ファイル名を省略してデータを読み込むと，一番先に見つかったデータファイルの先頭から読み始めます．

このままですと，どのデータファイルか確認ができないため，変数エリアの過不足が生じたり，全く違ったデータで処理を行なう可能性もあります．

ファイル名は，PUT，GETともに，付けるようにしてください．

6 VER ………マニュアルコマンド

カセットテープの録音状態を調べる

命令	VER "ファイル名"	読み方：ベリファイ
書式例	例1. VER "PRO−1" 例2. VER	語源：VERify
実行	RUN, WRTのいずれのモードでも実行可能	

6.1 VER命令

プログラムやデータをカセットテープに記録するには，記録方法に一定のルールがあります．これは，前述したとおりです．

VER命令は，はたしてルールどおりにプログラムやデータが記録されているかどうかを検査するコマンドです．

もし決められたルールどおり（所定のフォーマット）でなければ，「ERR9」を表示します．

所定のフォーマットで記録されていれば，……

PF：ファイル名

……を表示した後，検査が終われば，「READY P0」(RUN モードでプログラムエリアP0の場合）か「P 01………」（WRTモードの場合）の表示に変わります．

この命令を実行しても，PB−300のメモリーへプログラムやデータを読み込むわけではないので，プログラムエリアはどこでもかまいませんし，PB−300に記憶されているプログラムやデータへの影響はありません．

なお，ファイル名を省略すると，最初に見つけたファイル名のプログラムを検査します．

6.2 プログラムやデータが読み込めない場合

FA−3を使って，プログラムのLOAD，データのGETができない場合，次のような症状と原因が考えられます．

① 症状1：いつまでも" PF：" （LOAD Aなら"AF：", GETなら"VF：")というメッセージが出ないままテープが回り続ける．

〔原因〕＊再生時のテープレコーダ側のボリウムが低すぎる（最大にしても同じ症状の場合は，使用しているテープレコーダがFA−3との連動条件を満たしていないことになります）．

＊録音レベルが低すぎる．ただし，ヴァリアブルモニター（録音レベル自動設定機能）付のテープレコーダの場合は，これが原因ではありません．

② 症状2：ERR9というメッセージが出る．

〔原因〕＊FA−3の黒端子がテープレコーダの出力端子と確実につながっていない．

＊再生時のテープレコーダ側のボリウムが高過ぎる．

＊テープが痛んでいる．

＊テープレコーダのヘッドが汚れているため，機械が信号を読み取れない．

＊テープレコーダによっては，白端子と黒端子を同時に差し込んだとき，この症状が起こります．

このような場合は，記録または再生に必要な端子のみを接続して試してください．

12 便利なサブルーチン集

1 データ入力ルーチン

INPUTやKEY命令を活用するために，他のいろいろな命令や関数と組み合わせた入力ルーチンを紹介しておきます．

ルーチンは，サブルーチンで書かれたものと，プログラムの中の1行として書かれたものの2種類があります．

以下のルーチンの中で使っている行番号は仮のものですから，ご自分のプログラムに合わせて変更してください．

1.1 整数のみ変数に代入すればよい場合

```
100 INPUT "KAZU:", A:A=INT A
```

解説 入力した値を整数化して，再び元の変数に代入します．

1.2 正の整数のみ変数に代入すればよい場合

```
100 INPUT "KAZU:", A:A=INT (ABS (A))
```

解説 入力した値の絶対値を整数化して，元の変数に代入します．

絶対値の整数化でなければ，INT関数の性質上，負の数，たとえば−1.1は−2という具合に処理されます．この方がよいなら，A=ABS (INT(A)) としてください．

1.3 整数のみの入力

```
100 INPUT "KAZU:", A
110 IF FRAC A≠0 THEN 100
120 ……
```


解説 データを入力した変数の小数部だけをFRAC関数で取り出し，0でない（=小数部がある）場合は，再度データの入力(100行）へ戻ります．入力されたデータが整数なら，120行に書かれた命令を実行します．

1.4 0と正の数のみ入力

```
100 INPUT "KAZU:", A
110 IF A<0 THEN 100
120 ……
```

解説 データを入力した変数が，0より小さい場合（負の数）は，100行のデータの入力へ戻ります．

0か正の数であれば，120行の命令を実行します．

0か正の整数の場合は，前述の〈1.3〉の110行を〈1.4〉の120行として書いてください．

1.5 月（1月から12月）の入力

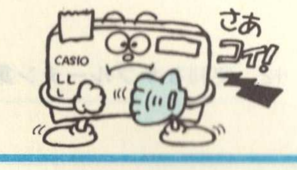
```
100 INPUT "TSUKI:", A:A=INT A
110 IF A≥1;IF A≤12 THEN 130
120 GOTO 100
130 ……
```

解説 データ（月）を入力した変数が，1から12の間に入っているかを110行で条件処理します．

1から12であれば，130行を実行します．

1から12以外の値であれば，再度100行のデータ入力を実行します．

100行で入力した値は整数化します（小数点の月はない）．



1.6 設定した範囲の数値だけ入力

```
1000 INPUT "SUU:", A
1010 IF A≥L; IF A≤G; RETURN
1020 GOTO 1000
```

解説　前述の〈1.5〉を汎用化したサブルーチンです．数値の範囲は，下限をL，上限をGにメインルーチンで代入します．

1.7 文字変数で7文字以下しか入力しない

```
100 INPUT "DATA", $
110 IF LEN ($) >7 THEN 100
120 A$=$
```

解説　入力命令の変数には$を使います．これなら30文字まで入力できます．

110行では，入力したデータが7文字を越えるかどうかを判定します．7文字を越えるなら，100行のデータ入力を再び実行します．

7文字以下なら，目的の文字変数に代入してプログラムを続けます．

1.8 文字変数の入力で，7文字までを有効とする入力方法

```
100 INPUT "DATA:", $:GOSUB 1000:A$=
    MID (1, L)
110 INPUT "DATA 2:", $:GOSUB 1000:B$
    =MID (1, L)
1000 L=LEN ($):IF L>7;L=7
1010 RETURN
```


解説　メインルーチンでの入力文とサブルーチンに分かれます．

メインルーチンでの入力の変数は$です．データを入力した後，サブルーチンで文字数を変数Lに代入して戻ります．

目的の文字変数には，最大の7文字までの文字列をMID関数で代入します．

サブルーチンでは，LEN関数でデータの文字数をLに代入します．そして，Lが7より大きいか判定します．7より大きければ7に設定します．この状態でメインルーチンへ戻ります．

この方法では，8文字より長い文字でも入力はできますが，文字変数には8文字目以降は記憶されません．

なお，最大7文字なので，MID（1，7）にすればよいのでは，とお考えになるかもしれません．

しかし，この方法では，$に記憶されている文字が1から6文字の場合には，エラー（ERR 2）が発生します．

1.9 数値入力でどのキーを押してもエラーが発生しない

（数値とするのは，0～9，−，．，E，Ē）

```
100 INPUT "KAZU:", $:GOSUB 1000:IF
    S=1 THEN 100
110 A=VAL ($)
 ⟆
1000 S=0:L=LEN ($)
1010 FOR I=1 TO L
1020 IF MID (I, 1) ≥ "0";IF MID (I, 1) ≤
     "." THEN 1060
1030 IF MID (I, 1) ="−" THEN 1060
1040 IF MID (I, 1) ≥"Ē";IF MID (I, 1) ≤
     "E" THEN 1060
```

```
1050　S=1：GOTO 1070
1060　NEXT I
1070　RETURN
```

変数名　A：数値を代入する変数
　　　　I：ループカウンタ
　　　　L：入力したデータの文字数
　　　　S：スイッチ．S=1では数値外の文字が入力されているので，データを再入力
　　　　$：データを入力する変数

解説　メインルーチンでは，データは$に入力します．

サブルーチンでそのデータが数値のみであるかを判定して，条件文で再入力か，次の行を実行するかを決めます．

サブルーチンは，文字が入っているかどうかのスイッチの変数を設定することから始まります．Sが0のままでメインルーチンへ戻れば，数値のみの入力です．

1文字ずつ，文字か数値かを検査するために，ループ命令を使っています．

1.10　特定のデータを入力したら，ループの外へ脱出

```
100 ……
 ⁝
150 INPUT "Data：", Z：IF Z=－999 THEN 200
 ⁝
200 ……
```

解説　データ入力の終わりを知るために使います．Data入力の最後は上の例では－999と決めます．このデータが入力されたら，入力は終了したので，計算などの別の処理へ移ります．

データの最後として設定する値は，入力するデータの中では発生する可能性のない値とします．0から－1000の間でデータを入力するには－999は使えません．

文字のデータでは，IF Z$="END" THEN 200 のようにします．

1.11　表示するデータについてのコメントの入力方法

```
 ⁝
100 PRINT Z$（I）；：INPUT "→Y/N", $
110 IF $="N" THEN 500
 ⁝
```

解説　PRINT命令で表示したデータについてのコメントを要求する場合に使います．

表示は，`TOKYO → Y／N?`（Z$(I) のデータがTOKYOのとき）となります．データに続いて，良いか悪いかを問うので，入力の判断がしやすくなります．

PRINT命令の最後を；にすると，続けてINPUT文の文字列を表示します．

データの入力時に，データのNo.を表示したい場合にも，この方法を使います．

```
100 FOR I=1 TO 10
110 PRINT "No."；I；：INPUT "␣Data：", Z(I)
120 NEXT I
```

……あるいは

```
110 PRINT "No."；I；"␣Data："；：INPUT Z(I)
```

表示は，`No.1 Data：?` となります．ここではIをデータのNo.として使っています．

1.12 キーを押すまで，プログラムの実行を中断する

```
  ⌇
100  A$=KEY : IF A$=" " THEN 100
  ⌇
```

解説　KEY入力を応用します．100行を実行すると，キーが押されない限り，変数A$はヌル(" ")で何も入っていません．

条件文で変数A$はヌルかどうかを判定します．ヌルなら再び100行を実行します．

キーがどれでもよいから押されたら，次の行を実行します．

たとえば，上の例に続いて……

```
110  IF A$="E" ; END        ……Eが押されたらプログラム終了
                              それ以外なら10行へ戻る
120  GOTO 10
```

または

```
110  IF A$="1" THEN 1000   ｝押されたキーによって
120  IF A$="2" THEN 2000   ｝分岐先を決める
130  GOTO 200
```

……のように使うことができます．

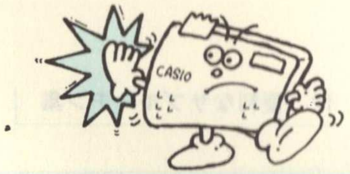
これらはよく使われるKEY命令の使用法です．

1.13 一定の回数だけKEY入力を実行する

```
1000  FOR I=1 TO 50
1010  A$=KEY
1020  IF A$=" " ; NEXT I
1030  RETURN
```

解説　ループによって，終値で設定した数だけKEY命令を実行します．

途中でキーが押されたら，メインルーチンへ戻ります．また，ループ命令を実行し終った場合も，変数A$はヌルのまま戻ります．



1.14 指定した文字数だけ，文字を入力するルーチン

```
  ⌇
100  L=5 : GOSUB 1000 : M$=$
  ⌇
1000  $=" "
1010  A$=KEY : IF A$=" " THEN 1010
1020  $=$+A$ : PRINT CSR 0 ; $ ;
1060  IF LEN ($) ≠L THEN 1010
1070  RETURN
```

変数名　A$：KEYで入力した文字
　　　　L：メインルーチンで指定した文字数
　　　　M$：メインルーチンでデータを代入
　　　　$：サブルーチンでデータを代入し，メインルーチンとサブルーチンのデータの受渡し役

解説　メインルーチンで文字数を指定し，サブルーチンを実行します．

サブルーチンからは，$に記憶されて戻ってくるので，目的の文字変数（M$）に代入します．

サブルーチンでは，KEY入力したデータをA$へ，A$は$へ代入します．$の文字数がメインルーチンで指定した文字数に達していなければ，再び1010行のKEY入力を実行します．

指定の文字数に達していたら，メインルーチンへ戻ります．

入力した文字は表示部に表示されます．

1.15 指定した桁数の数字（0～9）を入力する

（前項の〈1.14〉を数字用に変更したルーチンです）

```
  ⌇
100  L=5 : GOSUB 1000 : M=VAL($)
```

```
1000  $=" "
1010  A$=KEY : IF A$=" " THEN 1010
1020  IF A$≧"0" ; IF A$≦"9" THEN 1040
1030  GOTO 1010
1040  $=$+A$ : PRINT CSR 0 ; $ ;
1050  IF LEN ($) ≠L THEN 1010
1060  RETURN
```

解説　入力できるキーの制限と分岐命令を追加します．

0～9のキーのみ入力するので，1020行の条件文で，この範囲のキーだけが$に記憶されるようにします．

それ以外のキーは無効であるため，無視して再びKEY入力を実行します．

$に記憶された数字は，メインルーチンでVAL関数を使い数値に変換します．

2 データ表示，印字ルーチン

PRINT命令を使ったルーチンを紹介します．

以下のルーチンで使っている行番号や変数は仮のものです．ご自分のプログラムに合わせて変更してください．

2.1 表示をクリア ……プログラム中断

```
100  PRINT " "
```

解説　このPRINT命令は2つの動きをします．

① 前に実行したPRINT命令が；以外で終っていたら画面クリア，プログラム中断

② 前に実行したPRINT命令が；で終っていたら表示そのまま，プログラム中断

2.2 表示をクリア ……プログラム継続

```
100  PRINT CSR 0 ; "␣␣␣␣␣␣␣␣␣␣␣␣" ;
110  PRINT CSR 0 ;
```

解説　表示の先頭からスペースを12個表示してクリアします．

次の表示が先頭から始まるように，110行を実行します．

2.3 指定する表示位置から，指定する数だけ表示を消す

```
1000  PRINT CSR N ;
1010  FOR J=1 TO L
1020  PRINT "␣" ;
```

```
1030 NEXT J
1040 PRINT CSR N;
1050 RETURN
```

変数名 J：ループカウンタ
L：メインルーチンで指定するスペースの数（1～11）
N：メインルーチンで指定する消し始めるカーソル位置（0～11）

解説　このサブルーチンを実行する前のPRINT命令は;で終わっていることが必要です．;以外で終わっている場合は，表示がクリアされて，サブルーチンを実行します．

次にPRINT命令を実行するときのカーソル位置は，1040行で指定します．

2.4 プログラムの一時中断

```
100 PRINT "SET OK→EXE"
```

解説　PRINT命令は，;以外で終わることで，プログラムの実行を一時中断します．そこで，この機能を利用して，準備ができたら，EXEキーを押すことでプログラムを再開します．

これはINPUT命令の代わりに使います．表示するメッセージを工夫してください．

2.5 数値データの小数点表示，印字位置をそろえるサブルーチン

```
1000 L=4-INT (ABS (LOG (ABS D)))
1010 IF L=0 THEN 1030
1020 FOR J=1 TO L : PRINT "␣";: NEXT J
1030 PRINT D
```



```
1040 RETURN
```

```
 99999.999
  9876.543
   100.001
    50.5
     2.222
    -0.444
   -33.333
  -600.006
 -7000.007
-88888.888
```

変数　D：メインルーチンで代入された表示するデータ
L：表示開始位置
J：ループカウンタ

解説　1000行で表示する位置を計算します．4は最高5桁（99999）までのデータの場合です．目的のデータの範囲に合わせて変えてください．

このサブルーチンは，正，負いずれの数値にも使えます．

2.6 小数点の表示位置をそろえるサブルーチン……表示用

```
1000 L=4-INT (ABS (LOG (ABS D)))
1010 PRINT CSR L;D
1020 RETURN
```

解説　〈2.5〉のサブルーチンを表示部専用に作り変えたものです．1000行の4－を表示する数値の範囲，希望する位置に合わせて，0，1－から11－まで変えてください．

2.7 文字を右づめで表示，印字するサブルーチン

```
1000 L=12-LEN ($)
1010 IF L=0 THEN 1030
1020 FOR J=1 TO L : PRINT "␣";: NEXT J
1030 PRINT $
1040 RETURN
```

変数　L：表示，印字開始位置

```
TANUKI
KITUNE
INOSISI
NEKO
UMA
USAGI
```

J：ループカウンタ
$：メインルーチンで代入した
表示，印字する文字データ

解説　1000行で表示を開始する位置を計算します．印字する文字数によって12を変えてください．変数にしてメインルーチンで指定する使い方もあります．

2.8　文字を右づめで表示するサブルーチン……表示用

```
1000  L=12-LEN ($)
1010  PRINT CSR L;$
1020  RETURN
```

解説　〈1.7〉のサブルーチンを表示専用に変えたものです．

1000行の12は，右づめ最後が12桁目になります．用途に合わせて2から12の値に変えてください．

2.9　変数の記憶内容を変えずに，整数部のみを表示，印字

```
100  PRINT INT A
100  SET F0:PRINT A      } 小数点第1位で四捨五入して表示
100  PRINT RND (A, -1)   }
```

解説　3例とも整数部のみを表示します．

SET F0とRND（A，－1）では，正，負の符号に関係なく，小数点第1位を四捨五入して表示します．



2.10　表示を一時的に止めるサブルーチン……ループ命令

```
100  PRINT "KAME NOROI";
110  N=200:GOSUB 1000
 ⁞
1000  FOR J=1 TO N:NEXT J
1010  RETURN
```

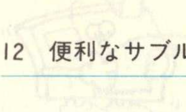
解説　メインルーチンでは，PRINT命令を；で終了します．

ループの終値＝停止時間をNに代入してサブルーチンを実行します．サブルーチンは，ループ命令を指定した数だけ実行します．

2.11　表示の一時停止のサブルーチン……ループ命令が使えない場合

```
100  PRINT "INU HOERU";
110  N=200:GOSUB 1000
 ⁞
1000  J=1
1010  IF J=N;RETURN
1020  J=J+1:GOTO 1010
```

解説　ループ命令のネスティングは4回までです．この制限を越える場合は，条件命令で同等の処理をします．



3 ディスプレイ活用ルーチン

………ゲームプログラムを作るために

ここでは，「ARARE & SENBEI」の12文字を，表示部でいろいろ動かししてみます.

ゲームプログラムを作るときの参考にしてください.

3.1 12文字の点滅

（ARARE & SENBEIの文字が表示部で点滅します）

① STOP キーを押せば止まる

```
100 PRINT CSR 0;"            ";
110 GOSUB 200
120 PRINT CSR 0;"ARARE&SENBEI";
130 GOSUB 200
140 GOTO 100
200 FOR J=1 TO 150:NEXT J
210 RETURN
```

② ループ命令の設定回数だけ点滅

```
100 $="ARARE&SEMBEI"
110 FOR I=1 TO 20
120 PRINT CSR 0;"            ";
130 GOSUB 200
140 PRINT CSR 0;$;
150 GOSUB 200
160 NEXT I
170 END
```





```
200 FOR J=1 TO 150:NEXT J
210 RETURN
```

3.2 左から順に１文字ずつ表示

① ■印が順に文字に変わる

```
100 $="ARARE&SENBEI"
110 PRINT CSR 0;"■■■■■■■■■■■■";
    :GOSUB 200
120 FOR I=1 TO 12
130 PRINT CSR (I-1);MID (I, 1);
140 GOSUB 200
150 NEXT I
160 GOTO 110
200 FOR J=1 TO 150:NEXT J
210 RETURN
```

② 点滅しながら文字を表示していく

```
100 $="ARARE&SENBEI"
110 FOR I=1 TO 12
120 PRINT CSR 0;"            ";
130 GOSUB 200
140 PRINT CSR 0;"■■■■■■■■■■■■";
150 PRINT CSR 0;MID (1, I);
160 GOSUB 200
170 NEXT I
180 END
200 FOR J=1 TO 150:NEXT J
210 RETURN
```

3.3　右から順に文字に変わる

①　■印が右から順に文字に変わる

```
100  $= "ARARE&SENBEI"
110  PRINT CSR 0;"■■■■■■■■■■■■";:
     GOSUB 200
120  FOR I=12 TO 1 STEP -1
130  PRINT CSR (I-1) ; MID (I, 1) ;
140  GOSUB 200
150  NEXT I
160  GOTO 110
200  FOR J=1 TO 150 : NEXT J
210  RETURN
```

②　右からARARE & SENBEIの順に表示

```
100  $="ARARE&SENBEI"
110  PRINT CSR 0;"■■■■■■■■■■■■";:
     GOSUB 200
120  FOR I=1 TO 12
130  PRINT CSR (12-I) ; MID (I, 1) ;
140  GOSUB 200
150  NEXT I
160  GOTO 110
200  FOR J=1 TO 150 : NEXT J
210  RETURN
```

```
120  FOR I=12 TO 1 STEP -1
130  PRINT CSR (I-1) ; MID (13-I, 1) ;
```

……でもよい．

③　点滅しながら文字を表示する（右から左へ）

```
100  $="ARARE&SENBEI"
110  FOR I=1 TO 12
120  PRINT CSR 0;"␣␣␣␣␣␣␣␣␣␣␣␣";
130  GOSUB 200
140  PRINT CSR 0;"■■■■■■■■■■■■";
150  PRINT CSR (12-I) ; MID (13-I, I) ;
160  GOSUB 200
170  NEXT I
180  END
200  FOR J=1 TO 150 : NEXT J
210  RETURN
```

3.4　文字が左から右へ流れる

１回目の表示（100〜140）

```
100  $="ARARE&SENBEI"
110  FOR I=1 TO 11
120  PRINT CSR 0; MID (13-I, I) ;
130  GOSUB 300
140  NEXT I
150  FOR I=0 TO 11
160  IF I=0 THEN 190
170  IF I=1; PRINT CSR 0;"␣"; :GOTO 190
180  PRINT CSR 0; MID (14-I, I-1) ;"␣";
190  PRINT CSR I; MID (1, 12-I) ;
200  GOSUB 300
210  NEXT I
220  GOTO 150
300  FOR J=1 TO 100 : NEXT J
310  RETURN
```



3.5　文字が右から左へ流れる

1回目の表示

```
100  $="ARARE&SENBEI"
110  FOR I=1 TO 11
120  PRINT CSR (12-I) ; MID (1, I) ;
130  GOSUB 300
140  NEXT I
```

2回目以後の表示

```
150  FOR I=0 TO 11
160  PRINT CSR 0 ; MID (I+1, 12-I) ;
170  IF I=0 THEN 210
180  PRINT "␣";
190  IF I=1 THEN 210
200  PRINT CSR (13-I) ; MID (1, I-1) ;
210  GOSUB 300
220  NEXT I
```

文字間にスペースを入れる

```
230  PRINT CSR 0 ; "␣" ; MID (1, 11) ;
240  GOSUB 300
250  GOTO 150
300  FOR J=1 TO 100 : NEXT J
310  RETURN
```

3.6　文字を1つ置きに2回に分けて表示

```
1回目    A␣A␣E␣S␣N␣E␣
2回目     R R & E B I
```

```
100  $="ARARE&SENBEI"
110  FOR  H=1  TO  2
120  FOR I=H TO 12 STEP 2
130  PRINT CSR (I-1) ; MID (I, 1) ;
140  GOSUB 200
150  NEXT I
160  NEXT H
170  END
200  FOR J=1 TO 150 : NEXT J
210  RETURN
```

練習問題の解答

〔練習問題 1〕（97 ページ）

```
10  G=0 : N=1          ……合計のGと件数のNを初期値にセット
20  INPUT "KAZU : ", K    ……数をKに入力
30  G=G+K          ……合計の計算
40  H=G/N          ……平均の計算
50  PRINT "GOKEI : " ; G    ……合計の表示
60  SET F1          ……小数点第 1 位までの表示を指定
70  PRINT "HEIKIN : " ; H    ……平均を表示
80  SET N          ……表示桁数の指定を解除
90  N=N+1          ……件数に1を加える
100 GOTO 20          ……20行へ分岐          （93ステップ）
```

〔練習問題 2〕（114 ページ）

```
10  INPUT "HANKEI : ", R    ……半径をRに入力
20  L=2＊π＊R          ……円周の計算
30  S=π＊R↑2          ……面積の計算
40  V=4/3＊π＊R↑3          ……体積の計算
50  MODE 7          ……プリンタを指定
60  PRINT "HANKEI␣: " ; R    ……半径の印字
70  SET F2          ……小数点第 2 位までの出力を指定
80  PRINT "ENSHUU ␣: " ; L    ……円周の印字
90  PRINT "MENSEKI : " ; S    ……面積の印字
100 PRINT "TAISEKI : " ; V    ……体積の印字
110 SET N          ……小数点第 2 位の指定を解除
120 MODE 8          ……プリンタ使用の解除
130 END          ……プログラムの終了          （138 ステップ）
```

〔練習問題 3〕（141 ページ）

```
10  N=0 : C=0 : D=0 : E=0 : F=0    ……各変数の初期値設定
20  PRINT N+1 ; "␣KYUUYO" ; : INPUT A    ……給与の入力
```



```
30  IF A=−999 THEN 200    ……入力の終わりは−999
40  INPUT "KOUJO", B    ……控除額の入力
50  C=A−B          ……支給額の計算
60  D=D+A          ……給与の総額計算
70  E=E+B          ……控除の総額計算
80  F=F+C          ……支給の総額計算
90  N=N+1          ……人数のカウント
100 MODE 7          ……プリンタの使用
110 PRINT "No." ; N    ……何人目かの印字
120 PRINT "␣␣␣␣KYUUYO : " ; A    ……給与の印字
130 PRINT "␣␣␣␣␣KOUJO : " ; B    ……控除額の印字
140 PRINT "␣␣␣ SHIKYUU : " ; C    ……支給額の印字
150 PRINT "ーーーーーーーーーーーーーーーーーーーーーー"    ……仕切り
160 MODE 8          ……プリンタの使用解除
170 GOTO 20          ……20行へ分岐
200 MODE 7          ……プリンタの使用開始
210 PRINT "SOUGAKU"    ……見出し（総額）の印字
220 PRINT "␣␣KYUUYO : " ; D    ……給与の総額印字
230 PRINT "␣␣␣KOUJO : " ; E    ……控除の総額印字
240 PRINT "␣SHIKYUU : " ; F    ……支給の総額印字
250 PRINT "␣␣NINZUU : " ; N    ……人数の合計を印字
260 MODE 8          ……プリンタの使用解除
270 END          ……プログラムの終了          （315 ステップ）
```

〔練習問題 4〕（158 ページ）

```
10  VAC          ……変数をすべてクリア
20  INPUT "CODE : ", C : C=INT C    ……区分コードを入力
30  IF C=99 THEN 200    ……99ならデータ印字へ
40  IF C≧1 ; IF C≦9 THEN 60    ……コードが1～9なら60行へ
50  GOTO 20          ……1～9以外なら20行へ
60  INPUT "SUURYOU : ", S    ……数量の入力
70  INPUT "TANKA : ", T    ……単価の入力
```

```
80  G=S*T  ……金額の計算
90  I(C)=I(C)+G  ……コード別に集計
100 GOTO 20  ……20行へ
200 MODE 7  ……プリンタの使用
210 FOR I=1 TO 9  ……配列I(1)～I(9)
220 PRINT "CODE";I;":￥";I(I)  ……コードと合計金額印字
230 NEXT I
240 MODE 8  ……プリンタ使用の解除
250 END  ……プログラムの終了
                                  (163ステップ)
```

〔練習問題5〕(189ページ)

```
10  PRINT "KEY WORD???"  ……タイトルの表示
20  GOSUB 200  ……乱数発生サブルーチンへ
30  PRINT CSR 0;"MOJI:";
40  INPUT $  ……A～Zの適当な文字列を入力
50  IF $=A$ THEN 120  ……キーワードと入力した文字が一致したら120行へ
60  L=LEN ($)  ……入力した文字列の長さを求める
70  FOR J=1 TO L  ……入力した文字数だけループを繰り返す
80  IF A$=MID (J, 1) THEN 110  ……入力した文字列の中にキーワードを含んでいたら110行へ
90  NEXT J  ……入力した文字列にキーワードが含まれていなければ次の行へ
100 PRINT CSR 0;"NG";:GOSUB 300:GOTO 30
110 PRINT CSR 0;"GOOD";:GOSUB 300:GOTO 30
120 PRINT "VERY GOOD→→KEY WORD:";A$
130 END
200 $="ABCDEF……………XYZ"       ┐
210 I=INT (RAN#*26)+1          │ A～Zの適当な1文字を乱
220 A$=MID (I, 1)              │ 数で発生させるサブルーチン
230 RETURN                     ┘
300 FOR K=0 TO 100:NEXT K      ┐ 表示時間をかせぐ待ちループサブルーチン
310 RETURN                     ┘
                              (251ステップ)
```


第3編

実用ライブラリー編

よい子悪い子

●ゲームコーナー●

何人かで競い合う一種のモグラタタキゲームです．最高得点者と最低得点者の名前が表示されるところがミソ．さてあなたはよい子？悪い子？普通の子？………



解説

表示部左はじから，0，1，2……9のナンバーがつけられています．ここへX印が，次々と現われますから，できるだけ速く0～9のキーを押します．

現われた位置のナンバーが表示中に正しくキータッチされればGD（Goodの略）が表示され，得点は＋10点，間違いもしくは時間切れの場合はNG（No Goodの略）が表示され，－10点となります．X印は全部で15回現われ，全部当たれば150点です．

PLAYが終ればランクがつけられます．まず得点が表示されます．次に名前の入力が出ればお楽しみ．最高点と最低点の獲得者は名前が登録され，ゲームの終了ごとに毎回，この名前が表示されます．最高点を出せば"よい子"，最低点は"悪い子"，その間は"普通の子"で残念ながら名前も得点も登録できません．さあチャレンジしてみましょう！

ゲームの表示 → ――X――――――――GD

↑↑↑↑↑↑↑↑↑↑ 当たればGD，はずれはNGを表示．この場合は2を押す．

キーの対応 → 0 1 2 3 4 5 6 7 8 9

ゲーム時使用キー → [0]～[9]

★ワンポイントアドバイス★

"X"の表示が速すぎてNGばかりで困る人は，プログラム320行の，FOR J＝1 TO 30 ←この数字を35～60にしてみましょう．なれたら元へ戻します．また，15回以上やりたい人は，70行の FOR I＝1 TO 15←この数字を好きな回数にセットしてください．200点，300点とれるかも……．

プログラム

プログラム入力前の設定

[MODE] [1] CLEAR[EXE]

[S] [DEFM ∨] [0] [EXE]

```
10 PRINT CSR 0;"**YWF GAME**";:GOSUB 300:IF Z$≠"" THEN 30
20 GOSUB 370:GOSUB 300:IF Z$="" THEN 10
30 PRINT CSR 0;"CLEAR→Y/N? ";:FOR J=1 TO 100:NEXT J
40 Z$=KEY:IF Z$="" THEN 40
50 IF Z$="Y";E=0:G=0:F$="---":H$=F$
60 A=0:PRINT CSR 0;"<GAME START>";:GOSUB 300
70 FOR I=1 TO 15:C=INT (RAN#*10)
80 GOSUB 380:PRINT CSR C;"X";
90 GOSUB 320:B=-10:K$="NG":IF VAL(D$)=C;B=10:K$="GD"
100 A=A+B:PRINT CSR 10;K$;:FOR J=1 TO 200:NEXT J
110 NEXT I
120 GOSUB 370:PRINT CSR 0;"TOKUTEN:";A
130 IF A>E;GOSUB 360:F$=MID(1,I):E=A
140 IF A≤G;GOSUB 360:H$=MID(1,I):G=A
150 PRINT "[YOIKO]",F$,"TOKUTEN:";E
160 PRINT "[WARUIKO]",H$,"TOKUTEN:";G
170 IF A≠E;IF A≠G;PRINT "FUTSUU NO KO","YOU","TOKUTEN:";A
180 PRINT "Try again→Y/N";
190 Z$=KEY:IF Z$="" THEN 190
200 IF Z$="Y" THEN 60
210 GOTO 10
300 FOR I=1 TO 40:Z$=KEY:IF Z$="";NEXT I
310 RETURN
320 FOR J=1 TO 30:D$=KEY:IF D$="";NEXT J
330 IF D$≥"0";IF D$≤"9" THEN 350
340 D$="10"
350 RETURN
360 INPUT "NAMAE:",$:I=LEN($):IF I>7;I=7
365 RETURN
*370 PRINT CSR 0;"           ";:RETURN
380 PRINT CSR 0;"----------- ";:RETURN
```

合計　695ステップ

メモリー内容	
A	ゲーム1回の得点
B	1トライの得点
C	X印の表示位置
D$	0～9のキー入力
E	最高得点
F$	最高得点者氏名
G	最低得点
H$	最低得点者氏名
I	ゲーム回数ループ
J	X表示長さループ
K$	GD，NGの表示
Z$	一般キー入力
$	氏名入力

＊本プログラム370行目のように，ダブルクォーテーション"　"で囲まれた文字列内の空白は，スペースキーを使って，正しい個数を入力します．

操作

	手順	キー操作	表示
	例	MODE 0	READY P0
	1	S PO	**YWF GAME**(点滅)
これまでの得点と氏名を消すならY，残すならNを押す→	2	数字か文字のいずれかのキーを押す	CLEAR→Y/N?
	3	(例) Y	〈GAME START〉
	4		−−−−−X−−−−
	5	(例) 5	−−−−−X−−−−GD
	6		−X−−−−−−−−
	7	(例) 2	−X−−−−−−−−NG
		15回繰り返し	⋮
ゲームの得点表示→	8		TOKUTEN： 50
	9	EXE	NAMAE：?
名前の入力．7文字までOK．→	10	(例)SUZUKI EXE	〔YOIKO〕
	11	EXE	SUZUKI
	12	EXE	TOKUTEN：50
	13	EXE	〔WARUIKO〕
誰もいない場合はこの表示．→	14	EXE	−−−
	15	EXE	TOKUTEN：0
ゲームを続けるならYを押す→	16	EXE	Try again→Y/N 以下 Yを押すと表示3から繰り返し

何回か回を重ね，最高と最低の間の得点者には，次の表示が出てきます．

	手順	キー操作	表示
手順13以降→	14	EXE	TANAKA
	15	EXE	TOKUTEN：−20
今までの最高か最低をオーバーできなかった人の表示→		EXE	FUTSUU NO KO
		EXE	YOU
		EXE	TOKUTEN：10 以下　手順16と同じ

テクテクゲーム

●ゲームコーナー●

PB−300とあなたがテクテクと歩きながら押し合いへしあいするゲームです．あなたのリズム感と反射神経が試されます．



解説

表示部左右から，矢印が相手側へ向かってどんどん進行します．左から進行するのはあなた．右から進行するのはPB−300です．点滅する信号に合わせてタイミングよくキーを押します．入力のタイミングが合えば，矢印はひとつ右へ進みます．コツは信号■がついた瞬間にキーを押すことです．そのときPB−300の矢印はひとつ後退します．

ボヤボヤしていたり，タイミングが合わなければ，PB−300の矢印が容赦なく右側から攻めてきます．

ワンゲームの終了は相手側に攻め込んだときです．これで1勝．ゲームはレベル1から4へだんだん強くなるので，自由に選んでください．ゲームの終了時にはこれまでの勝ち回数，負け回数と相手側への攻撃数が表示されます．攻撃数は，矢印が相手側へいくつ進んだかを数えているので，勝ち回数が多いからといって攻撃数が多いとは限りません．何回も押したり引いたりすれば，攻撃数が増えていきます．

ゲームの表示→ 〔■　→　←〕 ←ここまで押してくればあなたの勝ち！

あなたの矢印はキーにより右へ↑　↑PB−300の矢印は左へ進む

なお使用するキーは，特殊キー以外のどの文字，数字でも使えます．

★ワンポイントアドバイス★

負けがこむ人は50行L＝6−Lの数字の6を7〜15にすれば勝てるでしょう．もうひとつ．80，160，170，210行の“←”を“Ω”にして，310行のPRINT文を“You win((＊))”に，320行のPRINT文を“((＊))PB wins”にすると，宇宙大戦争になります．

プログラム

プログラム入力前の設定

MODE 1 CLEAR EXE

S DEFM(V) 0 EXE

```
10 PRINT "!Teku Teku!"
20 C=0:D=0:E=0:F=0
30 PRINT "You→ vs ←PB"
40 A=0:B=11:G=0
50 INPUT "LEVEL 1-4",L:L=INT L:L=6-L
60 PRINT "Start→any";
70 $=KEY:IF $="" THEN 70
*80 PRINT CSR 0;"→         ←";
100 GOSUB 400:GOSUB 500
110 IF G=2 THEN 200
120 IF $="" THEN 200
130 A=A+1:E=E+1:IF A=11;B=0:GOTO 300
140 PRINT CSR A-1;" →";
150 G=0:GOSUB 500:IF B=11 THEN 180
160 B=B+1:IF A=B-1;PRINT CSR B;"←";:GOTO 180
170 PRINT CSR B-1;"←";
180 GOTO 100
200 B=B-1:F=F+1:IF B=0;B=11:GOTO 300
210 PRINT CSR B;"← ";
220 G=0:GOSUB 500:IF A=0 THEN 250
230 A=A-1:IF A=B-1;PRINT CSR A;"→";:GOTO 250
240 PRINT CSR A;"→ ";
250 GOTO 100
300 PRINT CSR 0;
310 IF A>B;PRINT "You win";:C=C+1:GOTO 330
320 PRINT "PB wins";:D=D+1
330 FOR I=1 TO 300:NEXT I:PRINT CSR 0;
340 INPUT "REPLAY→Y/N ",$:IF $="Y" THEN 40
350 PRINT "You";C;" win";D;" lose"
360 PRINT "You";E;" vs";F;" PB"
370 END
400 PRINT CSR 0;"■";
410 FOR I=1 TO L:$=KEY:IF $≠"" THEN 440
420 NEXT I
430 FOR I=1 TO 5:NEXT I
440 PRINT CSR 0;" ";
450 IF A=0;PRINT CSR 0;"→";
460 RETURN
500 K$=KEY:IF K$="" THEN 520
510 G=G+1
520 RETURN
```

合計 680ステップ

メモリー内容	
A	You の 表示位置
B	PB の 表示位置
C	You の 勝回数
D	PB の 勝回数
E	You の 攻撃数
F	PB の 攻撃数
G	キーチェック回数
I	ループ
K$	キーチェック入力
L	ゲームのレベル
$	キー入力

*本プログラム80行目のように，ダブルクォーテーション" "で囲まれた文字列内の空白は，スペースキーを使って，正しい個数を入力します．

操 作

	手順	キー操作	表　示
		MODE 0	READY P0
タイトル表示→	1	S P0	!Teku Teku!
矢印の説明→	2	EXE	You→ vs ←PB
	3	EXE	LEVEL 1−4?
レベルを選んでスタート→	4	(例) 2 EXE	Start→any
	5	数字か文字のいずれかのキーを押す	(点滅) ■→　　　　←
	6	タイミングに合わせてキーを押す	■　　　　→　　←
		繰り返し	⋮
勝者の表示→	7		You win
			PB wins(PB−300が勝ちのとき)
	8		REPLAY→Y/N?
もし1度ならYを押す→	9	(例) Y EXE	LEVEL 1〜4? (手順3から繰り返す)

ゲームを続けないでNを入力すると次に続きます．

10		You 2 win 1 lose
11	EXE	You 33 vs 24 PB

電話料金の計算

●生活コーナー●

10円玉1個しか持っていない．3分間は話せる．それはわかっていても話しに夢中になって途中で切れてしまう……．よくありますね．10円で通話できる時間があと何秒残っているか，通話を始めてからの料金はいくらか，を刻々表示するのがこのプログラムです．



国立市　我妻利夫さん作成

解説

プログラムを実行すると，合計金額にこれまでの料金が記憶されていたら表示しますから，クリアするならY，ノーならNを入力します．

合計金額が0の場合か，クリアNの入力を行なうと，相手の氏名をFOR→？と聞いてきますから，氏名（7文字まで）を入力します．次に10円で何秒通話できるかを入力します．5秒なら5，3分なら3×60＝180 を入力します．小数点は切り捨てです．確認のため10円で何秒話せるか表示します．

ダイヤルして電話がつながったら EXE キーを押すと，10円で通話できる残り時間と料金をほぼ1秒毎に刻々と表示します．10円を超過すると次々と現在の料金が10円毎に累計されて表示されます．

通話が終われば，すぐに英数キーのいずれかを押すと，相手の氏名，通話時間と料金，これまでの合計金額が印字されます．各方面に電話をかける仕事には最適のプログラムです．秒数のカウントは110行のFOR文の初期値を変えることによって調整できます．

操作

	手順	キー操作	表　　示
		MODE 0	READY P0
合計が0以外のとき，クリアするか →	1	S 0	GOOKEI ¥ 300
	2	EXE	CLEAR→Y/N?
相手の氏名は？ →	3	（例）Y EXE	GOOKEI ¥:?
	4	（例）0 EXE	FOR→?（相手の氏名入力）
氏名を入力→	5	（例）IKEDA EXE	¥10=t sec?
10円で何秒か？→	6	（例）5 EXE	¥10=5 sec
通話中の表示→	7	EXE	5s ¥ 10 ～ 0s ¥ 10（秒数と累計金額を刻々と表示）
通話が終了したら→	8	英数キーのいずれか	プリンタで印字（印字例参照）
	9		CLEAR→Y/N?
	10	（例）N EXE	FOR→?
			（以下手順4から繰り返し）

プログラム

プログラム入力前の設定

MODE 1 CLEAR EXE

S DEFM/V 0 EXE

```
10 IF G≠0 THEN 250
20 INPUT "GOOKEI ¥:",G:G=INT G:B=1
25 INPUT "FOR→",A$
30 INPUT "¥10=t sec",T:T=INT T:R=10
40 PRINT "¥10=";T;" sec"
100 D=T
110 FOR I=33 TO 0 STEP -1
120 IF KEY≠"" THEN 200
130 NEXT I
140 D=D-1:GOSUB 300
150 IF D≠0 THEN 110
160 R=R+10:GOTO 100
200 MODE 7:PRINT "(";B;")";A$:PRINT R/10*T;"sec ¥";R:G=G+R
250 PRINT "GOOKEI ¥";G:B=B+1:MODE 8
260 INPUT "CLEAR→Y/N",$:IF $="Y" THEN 20
270 GOTO 25
300 PRINT CSR 0;"          ";
310 PRINT CSR 0;D;"s ¥";R;
320 RETURN
```

印字例

```
( 1)IKEDA
 50sec ¥ 50
GOOKEI ¥ 50
( 2)TAKANO
 36sec ¥ 120
GOOKEI ¥ 170
( 3)SAKAI
 26sec ¥ 130
GOOKEI ¥ 300
```

合計　314ステップ

メモリー内容	
A$	相手氏名
B	通話回数
D	ダウンカウンタ
G	合計料金
I	ループ
R	1回の通話料金
T	10円の通話時間
$	クリア入力

日数計算



●生活コーナー●

あなたの誕生日から，きょうまで何日過ぎたでしょう．基準になる日から計算したい日までの計算をするプログラムです．

解説

プログラムを実行すると，プリンタの使用を聞いてきます．結果をプリンタへ印字したい場合は，Yを入力します．

次に基準になる年（START NEN）を入力します．99までの年を入力すれば，その年号は昭和として計算します．1583*以上では西暦です．西暦で入力するか，昭和で入力するかなんてわずらわしい操作がいらないので，とてもスムースに入力できます．出力は入力したとおりに昭和（S）か西暦で出ます．

年の次は基準の月と日の入力です．13月とか2月30日なんてありえない月日は入力できません（但し2月29日のみは，3月1日として計算します）．

こんどは終りの年月日です．基準の年より前の年でも先の年でもかまいません．また，基準の年が西暦で入力したからといって，終りの年も西暦でという制限もありません．昭和でも西暦でも自由です．こちらも99までの年は昭和，100年以上は西暦と自動的に変換します．終りの月日も基準の月日と同じく，ありえない月日は入力できません．

ここまで入力すると，印字または表示します．数字の位置をそろえ，曜日も印字します．

なお，プログラム実行前に，必ずDEFM　12で変数エリアを拡張してください．そうしないとエラー表示が出ます．

★ワンポイントアドバイス★

現行の暦はグレゴリオ暦といい1582年10月にグレゴリオ13世が制定したものです．それ以前は128年で1日の誤差が出ていました．ですからこのプログラムでも1582年以前は受付けないようにしてあります．なおグレゴリオ暦は4年ごとのうるう年で誤差をなくし，さらに100の倍数でなおかつ400で割り切れる年はうるう年にしないことで正確な暦となっています．

プログラム

プログラム入力前の設定

[MODE] [1] CLEAR [EXE]

[S] [DEFM V] 12 [EXE]

```
10 PRINT "NISSU KEISAN":GOSUB 500
20 INPUT "PRTer→Y/N",Z$
100 Y$=W$:S=0:GOSUB 300:A=L:B=M:C=N:H=O:J=P:T=S
110 Y$=X$:S=0:GOSUB 300:D=L:E=M:F=N:I=O:K=P:U=S
200 IF Z$="Y";MODE 7:PRINT "--------------------"
210 Y$=W$:L=A:M=B:N=C:P=J:S=T:GOSUB 400
220 Y$=X$:L=D:M=E:N=F:P=K:S=U:GOSUB 400
230 PRINT "NISSU=";I-H;" nichi"
240 MODE 8:GOTO 100
300 PRINT Y$;" NEN";:INPUT L:L=INT L:IF L<100;L=L+1925:S=1
305 IF L<1583 THEN 300
310 PRINT Y$;" TSUKI";:INPUT M:M=INT M:IF M≥1;IF M≤12 THEN 320
315 GOTO 310
320 PRINT Y$;" HI";:INPUT N:N=INT N:G=Z(M):IF M=2;G=G+1
322 IF N≥1;IF N≤G THEN 330
325 GOTO 320
330 G=L-1:O=G*365+INT (G/4)-INT (G/100)+INT (G/400)+N
335 IF M=1 THEN 350
340 FOR Q=1 TO M-1:O=O+Z(Q):NEXT Q
350 IF L/4≠INT (L/4) THEN 380
360 IF L/100=INT (L/100);IF L/400≠INT (L/400) THEN 380
370 IF M>2;O=O+1
380 P=O-INT (O/7)*7
390 RETURN
400 PRINT Y$;:G=3:IF S=1;L=L-1925:G=1:PRINT " S";
410 G=G-INT (LOG L):IF G>0;PRINT " ";:G=G-1:GOTO 410
420 PRINT L;".";:IF M<10;PRINT " ";
430 PRINT M;".";:IF N<10;PRINT " ";
440 PRINT N
450 PRINT "    ";MID(P*3+1,3)
460 RETURN
500 W$="START":X$="END  ":$="SUNMONTUEWEDTHUFRISAT"
510 C=30:D=31
520 Z(1)=D:Z(2)=28:Z(3)=D:Z(4)=C:Z(5)=D:Z(6)=C:Z(7)=D:Z(8)=D
530 Z(9)=C:Z(10)=D:Z(11)=C:Z(12)=D:RETURN
```

プログラム	900ステップ
拡張メモリー	96ステップ
合計	996ステップ

メモリー内容		I	END日数	T	START年号スイッチ
		J	START曜日	U	END年号スイッチ
A	START年	K	END曜日	W$	"START" 表示
B	START月	L	年のワーク	X$	"END" 表示
C	START日	M	月のワーク	Y$	メッセージワーク
D	END年	N	日のワーク	Z$	プリンタ使用入力
E	END月	O	日数のワーク	Z(1)	
F	END日	P	曜日のワーク	∫	各月の日数
G	計算ワーク	Q	ループ	Z(12)	
H	START日数	S	年号スイッチ	$	曜日

印 字 例

```
--------------------
START 1955.  2.  2
                 WED
END   1983.  4.  1
                 FRI
NISSU= 10285 nichi
--------------------
START S 34.  9. 11
                 FRI
END   S 58.  7.  2
                 SAT
NISSU= 8695 nichi
--------------------
START S  5.  1.  5
                 SUN
END   1925. 11.  1
                 SUN
NISSU=-1526 nichi
```



円↔ドル換算

●ビジネスコーナー●

めまぐるしく変動する為替相場に十分対応できる銀行マン，証券マン向けのプログラムです．



解 説

出力は，円→ドルとドル→円の両方を表示します．したがって入力で円は，ドルは，と迷う必要はありません．

プログラムを実行すると，プリンタの使用とレート入力を聞いてきます．Ⓨを入力すればいずれも実行します．レートの入力は1＄何円かで入力してください．1回入力すれば，レートが変るまで入力の必要はありません．

プログラム

プログラム入力前の設定

[MODE] [1] CLEAR [EXE]

[S] [DEFM V] 0 [EXE]

```
 10 PRINT "Tsuuka K
    ansan"
 20 INPUT "PRTer→Y/
    N",P$
 30 INPUT "RATEinp→
    Y/N",$
 40 IF $="N" THEN 1
    00
 50 INPUT "$1=¥",Y
100 INPUT "Kingaku:
    ",K
110 IF P$="Y";MODE
    7
120 PRINT "RATE $1=
    ¥";Y
130 PRINT "¥";K;"=$
    ";RND(K/Y,-4)
140 PRINT "$";K;"=¥
    ";K*Y
150 MODE 8
160 GOTO 100
```

	表　　示
	Tsuuka Kansan
プリンタは？→	PRTer→Y/N?
レートは？→	RATEinp→Y/N?
金額は？→	Kingaku：?

見出し表示後，プリンタを使用するかどうかを聞いてきますが，Nを入力した場合は，ディスプレイに表示されます．

合計　180ステップ

メモリー内容	
K	換算金額
P$	プリンタ使用
Y	1ドルのレート
$	レート入力

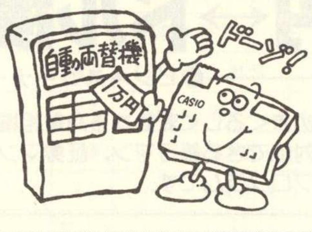
印字例

```
RATE $1=¥ 239.9
¥ 310=$ 1.292
$ 310=¥ 74369
RATE $1=¥ 239.9
¥ 1000=$ 4.168
$ 1000=¥ 239900
```

金種計算

●ビジネスコーナー●

経理マン向けの実用度ナンバーワンのプログラム．氏名も印字でき，「－999」入力で総合計も印字できます．



解説

金額を入力すると，その金額に必要なお金，例えば1万円札2枚，5千円札1枚……と表示してくれます．名前も同時に印字でき，誰あてかはっきり明示されるので便利です．

名前に「－999」を入力すれば，最初にそれまでに入力した金額の小計と人数，次に金額および総合計(K Total)を印字してプログラムは終了します．万一，間違った金額を入力した場合は，その金額にマイナスをつけて入力し，もう一度正しい金額を入れ直してください．なお，K Totalを印字すると終了ですが，続けたい人は260行のENDをGOTO40にしてください．

印字例

```
--------------------
AOKI TAROU sama
  ¥ 256749
--------------------
¥ 10000-→ 25mai
¥  5000-→ 1mai
¥  1000-→ 1mai
¥   500-→ 1mai
¥   100-→ 2mai
¥    50-→ 0mai
¥    10-→ 4mai
¥     5-→ 1mai
¥     1-→ 4mai
--------------------
```

```
--------------------
BABA YASUKO sama
  ¥ 185093
--------------------
¥ 10000-→ 18mai
¥  5000-→ 1mai
¥  1000-→ 0mai
¥   500-→ 0mai
¥   100-→ 0mai
¥    50-→ 1mai
¥    10-→ 4mai
¥     5-→ 0mai
¥     1-→ 3mai
--------------------
```

```
--------------------
TOTAL ¥ 441842
Ninzuu: 2nin
--------------------
¥ 10000-→ 43mai
    s kei ¥ 430000
¥  5000-→ 2mai
    s kei ¥ 10000
¥  1000-→ 1mai
    s kei ¥ 1000
¥   500-→ 1mai
    s kei ¥ 500
¥   100-→ 2mai
    s kei ¥ 200
¥    50-→ 1mai
    s kei ¥ 50
¥    10-→ 8mai
    s kei ¥ 80
¥     5-→ 1mai
    s kei ¥ 5
¥     1-→ 7mai
    s kei ¥ 7
--------------------
K Total ¥ 441842
```


プログラム

プログラム入力前の設定

[MODE] [1] CLEAR [EXE]

[S] [DEFM V] [1] [EXE]

```
  10 PRINT "Kinshu C
     al."
  20 VAC
  30 GOSUB 1000
  40 INPUT "PRTer→Y/
     N",X$
 100 INPUT "Namae:",
     $:IF $="-999" T
     HEN 200
 110 GOSUB 300
 120 INPUT "Kingaku:
     ",A:Z(1)=SGN A
 125 B=B+A:C=C+1*Z(1
     ):IF Z(1)=-1 TH
     EN 150
 130 GOSUB 500:PRINT
      $;" sama","¥
     ";A:GOSUB 500
 140 $="    "
 150 FOR W=1 TO 9
 160 D=INT (ABS A/D(
     W))*SGN A:M(W)=
     M(W)+D:A=A-D*D(
     W)
 165 IF Z(1)≧0;GOSUB
      600
 170 NEXT W
 175 IF Z(1)=-1 THEN
      100
 180 GOSUB 500:PRINT
      :PRINT :MODE 8
 190 GOTO 100
 200 GOSUB 500:PRINT
      "TOTAL ¥";B,"N
     inzuu:";C;"nin"
     :GOSUB 500
 210 A=0:$="    "
 220 FOR W=1 TO 9
 230 Y=D(W)*M(W):A=A
     +Y:D=M(W):GOSUB
      600:PRINT $;"s
      kei ¥";Y
 240 NEXT W
 250 GOSUB 500:PRINT
      "K Total ¥";A
 260 MODE 8:END
 300 W=LEN($):IF W>1
     5;W=15
 310 $=MID(1,W):RETU
     RN
 500 IF X$="Y";MODE
     7:PRINT "------
     --------------"
 510 RETURN
 600 Z=4-INT (LOG D(
     W))
 610 PRINT "¥";:IF Z
     ≠0;PRINT MID(1,
     Z);
 620 PRINT D(W);"-→"
     ;D;"mai"
 630 RETURN
1000 D(1)=10000:D(2)
     =5000:D(3)=1000
     :D(4)=500:D(5)=
     100:D(6)=50
1010 D(7)=10:D(8)=5:
     D(9)=1
1020 RETURN
```

プログラム	654ステップ
拡張メモリー	8ステップ
合計	662ステップ

メモリー内容	
A	入力金額，総合計
B	入力金額トータル
C	人数合計
D	金種計算枚数，金額
D(1) ～ D(9)	金種別金額
M(1) ～ M(9)	金種別枚数
W	ループ，文字数
X$	プリンタ使用入力
Y	小計
Z	プリントフォーマット
Z(1)	訂正フラグ
$	氏名

操作

	手順	キー操作	表　示
入力前の設定でメモリーを１つ増設することに注意→		S DEFM(V) 1 EXE	＊＊＊VAR：27
		MODE 0	READY P0
	1	S 0	Kinshu Cal.
プリンタを使用するか？→	2	EXE	PRTer→Y/N?
	3	(例) Y	Namae：?
15文字までOK→	4	(例)AOKI TAROU EXE	Kingaku：?
	5	(例)256749 EXE	プリンタから印字されます．(印字例参照) 手順３でNを押せば，ディスプレイ表示となります．
			Namae：?
		(例)BABA YASUKO EXE	以下手順3〜5の繰り返し
			Namae：?
総合計を出すときは−999→	6	−999 EXE	TOTAL ￥ 416221
	7		以下プリンタから印字して終了

★ワンポイントアドバイス★

金額を誤って入力した場合は，いったんマイナスでその金額を入力し，あらためて正しいものを入れてください．マイナス金額の入力の場合，訂正を行なうだけで印字はせず，手順３にもどります．

パソコン・レジスタ

●ビジネスコーナー●



レシートの発行と，売上げ数および金額の集計を行なう本物のレジスタ以上にすばらしいパソコンレジスタです．１回に登録できる商品数は20個までです．

このプログラムは，カセットインタフェイスFA−3が必要です．プログラムは２本に分かれており，「データのメンテナンス」と「レシートの発行・集計」の２本のテープを使い分けます．まず営業を始める前に「データのメンテナンス」で，その日の準備をします．

解説１

カセットインタフェイス　FA−3が必要

■ データのメンテナンス

商品名と単価の登録をするプログラムです．まずプログラムを入力する前に必要なことは，各変数の初期値設定をプログラムの中で行なっていないので，必ずVAC EXE を実行して変数をクリアしてください．メモリーの拡張はDEFM 43 EXE です．

このプログラムは，

1：Tape INPUT →テープからデータを読む．
2：KEY INPUT →キーからデータを入力する．
3：Teisei →データを訂正する．
4：Sakujo →データの削除．
5：Data List →データのリスト印字．
6：Data SAVE →データをテープに記録する．

の６つの機能を持っており，毎日，単価や品目が変わる業務には便利なものといえます．

カセットレコーダとカセットインタフェイスFA−3をPB−300にまず接続します．それぞれの処理はメニューの表示が出れば，数字キー１〜６で指定できますが，P１からP６のエリアを S 1〜6 で直接指定することもできます．処理が終ればメニューへ戻ります．

商品名と単価の新規入力と追加入力は２を指定します．商品名は30文字まで

入りますが，登録は先頭の7文字です．

データを入力したら5を指定してリストをとり，確認します．訂正があれば3を，削除があれば4を指定します．

データの入力，訂正，削除が終ると，テープに記録するかどうかを聞いてきますから，データが完成したらYを入力してテープに記録してください．データの記録が終わるか，Nを入力すれば，メニュー表示に戻ります．このデータを記録したテープは，プログラムのP1－20行のGET "R－DATA" とP6－20のPUT "R－DATA" のひとつのファイル名で入出力しますから，1本のテープに異なるデータを連続的に記録するには，必ずデータ間を広くあけて「0～8野菜　16～24果物　32～40パン……」のようにカウンタでテープの記録位置をメモしておかないと不便です．できれば，データ1本にテープ1本が望ましいでしょう．なお「データのメンテナンス」のプログラムのマスターテープも必要です．その場合，プログラムエリアをP1～P9まで使っているので，テープへ記録するには「SAVE A」，読み出すには「LOAD A」のコマンドで行ないます．

プログラム1

■データのメンテナンス

プログラム入力前の設定

MODE 1 CLEAR A EXE

VAC EXE S DEFM(V) 43 EXE

```
P0
10 PRINT "1:Tape INPUT";:A=20:GOTO 70
20 PRINT "2:KEY INPUT";:A=30:GOTO 70
30 PRINT "3:Teisei";:A=40:GOTO 70
40 PRINT "4;Sakujo";:A=50:GOTO 70
50 PRINT "5:Data List";:A=60:GOTO 70
60 PRINT "6:Data SAVE";:A=10
70 GOSUB 1000:IF $≠"" THEN 100
80 GOSUB 1000:IF $≠"" THEN 100
90 GOTO A
100 IF $≧"1";IF $≦"6";GOTO #VAL($)
110 GOTO 10
1000 FOR D=1 TO 30:$=KEY:IF $="";NEXT D
1010 PRINT CSR 0;" ";:PRINT CSR 0;:RETURN

P1
10 PRINT "Tape INPUT","SET OK→EXE"
20 GET "R-DATA"E,Z(20)
30 GOTO #0

P2
10 PRINT "Key INPUT":B=E
20 B=B+1:IF B>20 THEN 50
30 A=1:GOSUB #7:IF $="-999" THEN 50
40 GOSUB #8:GOSUB #9:C=1:GOTO 20
50 E=B-1:IF C=1 THEN #6
60 GOTO #0

P3
10 PRINT "Teisei"
20 INPUT "Data No.→",B:IF B>E THEN 60
30 A=0:GOSUB #7
40 INPUT "Teisei→Y/N",$:IF $="N" THEN 20
50 A=1:GOSUB #7:GOSUB #8:GOSUB #9:C=1:GOTO 20
60 IF C=1 THEN #6
70 GOTO #0

P4
10 PRINT "Sakujo"
20 INPUT "Data No.→",B:IF B>E THEN 70
30 A=0:GOSUB #7:INPUT "Sakujo→Y/N",$
40 IF $="N" THEN 20
50 FOR D=B TO E-1:F$(D)=F$(D+1):Z(D)=Z(D+1):NEXT D:C=1
60 E=E-1:GOTO 20
70 IF C=1 THEN #6
80 GOTO #0

P5
10 MODE 7:PRINT "Data List"," No. Hinmei  Tanka"
20 FOR D=1 TO E:IF D<10;PRINT " ";
30 PRINT D;" ";F$(D);" ";Z(D):NEXT D:MODE 8:GOTO #0

P6
10 INPUT "Data SAVE→Y/N",$:IF $≠"Y" THEN 30
20 PRINT "SET OK→EXE":PUT "R-DATA"E,Z(20):C=0
30 GOTO #0

P7
10 PRINT "No.";B;"Hinmei→";:IF A=1;INPUT $:RETURN
20 PRINT F$(B):RETURN

P8
10 D=LEN($):IF D<7;$=$+" ":GOTO 10
20 F$(B)=MID(1,7):RETURN

P9
10 PRINT "No.";B;"Tanka→";:INPUT Z(B):RETURN
```

プログラム	973ステップ
拡張メモリー	344ステップ
合計	1317ステップ

メモリー内容	
A	分岐先，品名表示のスイッチ
B	データ件数のワークエリア
C	データ記録スイッチ
D	ループカウンタ
E	データ件数
$	各種入力
F$(1)〜F$(20)	商品名
Z(1)〜Z(20)	単価

操作1

■ データのメンテナンス

	手順	キー操作	表　　示
変数エリアの拡張→		VAC [EXE] [S] [DEFM V] 43 [EXE]	＊＊＊ VAR : 69
	1	[S][0]	1 : Tape INPUT ↓　点滅繰り返し 6 : Data SAVE
[2]データ入力 まずデータ入力→	2	(例) [2]	Key INPUT
あいている番号から聞いてきます→	3	[EXE]	No. 1Hinmei→?
次に単価は――→	4	(例)TAIYAKI [EXE]	No. 1 Tanka→?
	5	(例) 90 [EXE]	No. 2 Hinmei→? 次々とNo 20まで繰り返す
カセットに記録する？――→	6		Data SAVE→Y/N?
FA−3にカセットレコーダをセットする	7	(例) [Y] [EXE] (Nにすれば表示1へ)	SET OK→EXE (カセットは録音にする)
	8	[EXE]	メニュー表示に戻って完了
[3]データ訂正	1	(例) [3]	Teisei
	2	[EXE]	Data No. →?
すでに入力したデータNo.で訂正する→	3	(例) [5] [EXE]	No. 5 Hinmei→ONIGIRI
	4	[EXE]	Teisei→Y/N?
	5	(例) [Y] [EXE]	No. 5 Hinmei→?
	6	(例)COFFEE [EXE]	No. 5 Tanka→?
終了の場合は21以上の数を入力すればよい――→	7	(例) 200	Data No. →?
	8	(例) 21	Data SAVE→Y/N?
		[N] [EXE]	メニュー表示に戻って完了
[4]データ削除	9	(例) [4]	Sakujo
	10		Data No. →?
	11	(例) [1]	No. 1 Hinmei→TAIYAKI

なおデータを削除した場合は，FOR～NEXT 文によって，次々とNo.順に商品名が繰り上って削除番号を埋めていきますから，必ずData SAVE と LIST をとるようにしましょう．

	手順	キー操作	表　　示
	12	[EXE]	Sakujo→Y/N?
	13	(例) [Y] [EXE]	Data No. →?
	14	(例) 33 [EXE]	Data SAVE→Y/N?
		[N] [EXE]	メニュー表示に戻って完了
[5]Data List	1	(例) [5]	プリンタによってデータ出力 (印字例参照)
[6]テープ入力	1	(例) [6]	Data SAVE→Y/N?
	2	[Y] [EXE]	SET OK→EXE
	3	[EXE]	
			メニュー表示に戻って完了 (点滅繰り返し)

Data List 印字例

```
Data List
 No. Hinmei   Tanka
  1 TAIYAKI    70
  2 CAKE       470
  3 TEA        320
  4 ICE CRE    150
  5 COFFEE     200
  6 POTATO     180
  7 HOT DOG    150
  8 APPLE      135
  9 ORANGE     50
 10 CHICKEN    345
 11 POPCORN    120
 12 MILK       60
 13 BEER       375
 14 WHISKY     690
 15 COLA       100
 16 WINE       560
 17 CHEESE     120
 18 CHOCO      175
 19 CIGARET    185
 20 RICE       90
```

★ワンポイントアドバイス★

このパソコンレジスタでは，データをテープで記録しておくので，データテープ，プログラムテープはそれぞれ２本ずつバックアップしておくと，万一，テープを１本消去しても安心です．

また，レシート発行のプログラムで，商品番号を入力すると，いちいち商品名が確認のために表示され「→Y＝0」の入力を要求しますが面倒な人，商品番号入力に自信のある人は，次のようにしてください．

160行のIF～THEN180を190 に変える．

180行は，全部消去する．

どうですか？ 前より速くなって，実際上，使いやすいのではないでしょうか？

解説2

■ **レシート発行・集計**

レシートへは，メッセージと発行日，個別の商品名，数量，単価と金額，それに合計の金額が印字されます（印字例1）.

誤った商品をレシートへ印字したら，もう一度同じ商品を指定して－の数量を入力します．赤伝処理も同様に－の数量を入力します．合計の数量，金額も同時に加減されます（印字例2）.

レシートの発行時には，データ登録されたNo.を入力します．登録した商品名が表示されますから，それで間違いなければ0を入力し，次に数量を入力します．100を入力すれば，登録外の商品が個別入力できます．商品名（7文字まで），単価，数量を入力します（印字例3）.

また，99を入力すれば，それまでの入力を〆めて合計を計算，印字します．

集計は，登録した商品別に，商品名，数量の合計，単価，金額の合計，登録外商品の数量合計，金額の合計，そして全体の合計金額をプリントします（印字例4）.

レシート印字例1

```
THANK YOU!
SNACK,FOODS no PINKY
     1983. 4. 1
--------------------
CAKE          5 @ 470
                ♦ 2350
TEA          10 @ 320
                ♦ 3200
COFFEE        8 @ 200
                ♦ 1600
HOT DOG      12 @ 150
                ♦ 1800
POPCORN      14 @ 120
                ♦ 1680
BEER          5 @ 375
                ♦ 1875
COLA         23 @ 100
                ♦ 2300
--------------------
TOTAL         ¥ 14805
```

レシート印字例2

```
THANK YOU!
SNACK,FOODS no PINKY
     1983. 4. 1
--------------------
WINE         20 @ 560
               ♦ 11200
WINE        -20 @ 560
               ♦-11200
CHEESE        5 @ 120
                 ♦ 600
BEER         12 @ 375
                ♦ 4500
CIGARET       8 @ 185
                ♦ 1480
CIGARET      -5 @ 185
                 ♦-925
WHISKY        5 @ 690
                ♦ 3450
--------------------
TOTAL          ¥ 9105
```

レシート印字例3

```
THANK YOU!
SNACK,FOODS no PINKY
     1983. 4. 1
--------------------
SUSHI        5 @ 1200
                ♦ 6000
TENDON      13 @ 800
               ♦ 10400
UNADON      21 @ 2000
               ♦ 42000
SASHIMI      4 @ 5000
               ♦ 20000
SAKE        50 @ 400
               ♦ 20000
TSUMAMI     11 @ 700
                ♦ 7700
ETC.        1@ 25300
               ♦ 25300
--------------------
TOTAL        ¥ 131400
```


「レシート発行・集計」プログラムは，データをテープから入力する処理，商品名の入力，検索，例外処理，数量入力と印字，集計処理，数量のクリアに分かれます．主なポイントを順に説明しましょう．

(1) テープ入力

メモリーに商品名と単価が登録されていなければデータの入力を要求します.「データのメンテナンス」で作ったデータテープから入力してください．

(2) データ入力とレシート印字

No.→？と聞いてきたら，Data list（登録の商品番号）を見ながら，その番号を入力します．すると商品名が表示され，Yesなら0を押します．

Suu→？で数量を入力します．すると，商品名，数量，単価，小計を印字します．この繰り返しとなります．

(3) 合計と集計の出し方

No.→？と聞いてきたら，99を入力します．商品が1個以上，レシートに印字されていたら，そのレシートをしめて，合計金額を印字します．

商品が1個も印字されていないとき，つまり最初のNo.→で99を入力すれば，それまで入力されたデータのすべてを集計し印字します．印字例4を参照してください．

(4) 登録外商品の入力とメッセージ

No.→？のとき100を入力するとETC？と表示されますから，商品名（7文字まで），単価，数量を入力します．

またレシートの頭に印字されるメッセージは，プログラムの120行と130行のPRINT 文を変更してください．特に日付は毎日，始業前に変更確認しましょう．

レシート印字例4

```
THANK YOU!
SNACK,FOODS no PINKY
     1983. 4. 1
--------------------
TAIYAKI       5 @ 70
                 ♦ 350
CAKE          8 @ 470
                ♦ 3760
TEA          15 @ 320
                ♦ 4800
ICE CRE       6 @ 150
                 ♦ 900
COFFEE        0 @ 200
                   ♦ 0
POTATO       25 @ 180
                ♦ 4500
HOT DOG      19 @ 150
                ♦ 2850
APPLE         2 @ 135
                 ♦ 270
ORANGE       20 @ 50
                ♦ 1000
CHICKEN      42 @ 345
               ♦ 14490
POPCORN      35 @ 120
                ♦ 4200
MILK          3 @ 60
                 ♦ 180
BEER          6 @ 375
                ♦ 2250
WHISKY       10 @ 690
                ♦ 6900
COLA         68 @ 100
                ♦ 6800
WINE          0 @ 560
                   ♦ 0
CHEESE        0 @ 120
                   ♦ 0
CHOCO         6 @ 175
                ♦ 1050
CIGARET       1 @ 185
                 ♦ 185
RICE          7 @ 90
                 ♦ 630
ETC.          6 @ 0
                ♦ 9000
--------------------
TOTAL         ¥ 64115
```

プログラム2

プログラム入力前の設定

[MODE] [1] CLEAR A [EXE]

[S] [DEFM/V] 43 [EXE] VAC [EXE]

```
P0
 10 IF E=0;GOTO #4
100 A=0
110 PRINT "PASOCON
    REGI"
120 MODE 7:PRINT "T
    HANK YOU!","SNA
    CK,FOODS no PIN
    KY"
130 PRINT "   198
    3. 4. 1","-----
    ------------
    ":MODE 8
140 INPUT "No.→",D:
    D=INT D:IF D=99
     THEN 300
150 IF D=100;D=21:I
    NPUT "ETC.",B$:
    INPUT "ETC.Tank
    a→",F
155 GOTO 200
160 IF D>0;IF D≦E T
    HEN 180
170 GOTO 140
180 PRINT F$(D);"→Y
    =0";:INPUT $:IF
    $≠"0" THEN 140
190 B$=F$(D):F=Z(D)
200 INPUT "Suu→",C:
    Z(20+D)=Z(20+D)
    +C
210 A=A+F*C:IF D=21
    ;Z(42)=Z(42)+F*
    C
220 MODE 7:Z(43)=C*
    F:GOSUB #3:MODE
     8:F=-100
230 GOTO 140
300 IF F=-100;F=0:G
    OTO #2
310 GOTO #1
```

```
P1
 10 MODE 7
 20 FOR D=1 TO E
 30 A=A+Z(D)*Z(D+20
    )
 40 B$=F$(D):C=Z(D+
    20):F=Z(D):Z(43
    )=C*F:GOSUB #3
 50 NEXT D:A=A+Z(42
    )
 60 B$="ETC.    ":C=
    Z(41):F=0:Z(43)
    =Z(42):GOSUB #3
 70 MODE 8:GOTO #2

P2
 10 MODE 7
 20 PRINT "--------
    ------------"
 30 $="
    ":PRINT "TOTAL"
    ;
 40 IF A=0;PRINT $;
    :GOTO 60
 50 PRINT MID(1,12-
    INT (LOG ABS (A
    )));
 60 PRINT "¥";A
 70 PRINT "","",""
 80 MODE 8:GOTO #0

P3
 10 $="    "
 20 PRINT B$;:IF C=
    0;PRINT $;:GOTO
     50
 30 IF LOG (ABS C)≧
    4 THEN 50
```

```
 40 PRINT MID(1,4-I
    NT (LOG (ABS C)
    ));
 50 PRINT C;:IF F=0
    ;PRINT $;:GOTO
    80
 60 IF LOG (ABS F)≧
    4 THEN 80
 70 PRINT MID(1,4-I
    NT (LOG (ABS F)
    ));
 80 PRINT "@";F
100 $="
       "
110 IF Z(43)=0;PRIN
    T $;:GOTO 130
120 PRINT MID(1,17-
    INT (LOG (ABS Z
    (43))));
130 PRINT "*";Z(43)
140 RETURN

P4
 10 PRINT "Tape INP
    UT","SET OK→EXE
    "
 20 GET "R-DATA"E,Z
    (20)
 30 GOTO #0

P9
 10 FOR D=21 TO 42:
    Z(D)=0:NEXT D:E
    ND
```

プログラム	947ステップ
拡張メモリー	344ステップ
合計	1291ステップ

メモリー内容		E	データの件数	Z(21)～Z(40)	数量
		F	単価		
A	金額の合計	$	各種入力		
B$	検索品名，印字品名	F$(1)～F$(20)	商品名	Z(41)	個別入力の数量
C	ループの初期値 数量の入力			Z(42)	個別入力の金額
D	ループ，入力した 品名の文字数	Z(1)～Z(20)	単価	Z(43)	合計印字のワーク

なお,「レシート発行・集計」のプログラムをテープに記録するには,「SAVE A」, 読み出しには「LOAD A」のコマンドで行ないます.

操作2

●データをテープから読み出すとき

	手順	キー操作	表　示
	1	[S] [DEFM/V] 43 [EXE]	＊＊＊VAR：69
	2	[S] [4]	Tape INPUT
	3	[EXE]	SET OK→EXE
カセットをセットしたら[EXE]──→	4	[EXE]	VF：R−DATA

●登録商品のレシート発行

	キー操作	表示
	[S] [0]	PASOCON REGI
	[EXE]	（店名と日付を印字）
		No.→?
（例）[5]	[EXE]	COFFEE→Y＝0?
（例）0	[EXE]	Suu→?
（例）10	[EXE]	（レシートに印字） No.→?（繰り返し）

●登録外商品の入力──個別入力

個別入力は100，合計と集計は99と覚えましょう．

		表示
		No.→?
100	[EXE]	ETC.?
（例）TENDON	[EXE]	ETC. Tanka→?
（例）650	[EXE]	Suu→?
（例）20	[EXE]	（レシート印字）

減価償却額の計算

—●ビジネスコーナー●—

解説

減価償却額の計算には，定額法と定率法があります．

定額法の年度償却額は

年度償却額＝（取得価額－残存価額）÷耐用年数

で計算し，毎年一定額です．

定率法では，

年度償却額＝残額×償却率

$$償却率 = 1 - \sqrt[n]{\frac{残存価額}{原価}} = 一定 \qquad n：耐用年数$$

で計算します．償却率が毎年一定です．

プログラムでは，取得金額，残存価額，耐用年数を入力することで，定額法と定率法の２通りで計算します．出力は各年の年度償却額，償却額の累計，残額です．

操作

	手順	キー操作	表示
		MODE 0	READY P0
	1	S 0	Genka Shoukyaku
	2	EXE	Shutoku ¥?
取得価額入力→	3	(例)1200000 EXE	Zanzon ¥?
残存価額入力→	4	(例) 90000 EXE	Taiyou Nen : ?
耐用年数入力→	5	(例) 7 EXE	PRTer→Y/N
プリンタ使用？→	6	(例) Y EXE	（プリンタへ印字）
			（Nならディスプレイ表示）

なおプリンタ印字の場合，耐用年数が長い場合は，プリンタロール紙を多く使いますから，充分残っているか必ず確認しましょう．

プログラム

プログラム入力前の設定

MODE 1 CLEAR EXE

S DEFM(V) 0 EXE

```
10 PRINT "Genka Shoukyaku"
20 INPUT "Shutoku ¥",X:X=INT (ABS (X))
30 INPUT "Zanzon ¥",Z:Z=INT (ABS (Z))
40 INPUT "Taiyou Nen:",N:N=INT (ABS (N))
50 INPUT "PRTer→Y/N",$:IF $="Y";MODE 7
60 $="    ¥"
70 PRINT "< GENKA SHOUKYAKU >":GOSUB 300
80 PRINT "Shutoku Kagaku",$;X
90 PRINT "Zanzon Kagaku",$;Z
100 PRINT "Taiyou nensuu";N;"nen":GOSUB 300
110 PRINT "* TEIGAKU HOU *"
120 S=INT ((X-Z)/N):T=0
130 FOR I=1 TO N:G=X-S*I
140 IF G<Z THEN 160
150 T=T+S:GOSUB 350
160 NEXT I:GOSUB 300
170 PRINT "* TEIRITSU HOU *"
180 R=1-(Z/X)↑(1/N):R=RND(R,-4)
190 G=X:T=0
200 PRINT "Shoukyaku Ritsu:",R*100;"%"
210 FOR I=1 TO N:S=INT (G*R):T=T+S:G=G-S
220 IF G<Z THEN 240
230 GOSUB 350
240 NEXT I:GOSUB 300
250 MODE 8
260 GOTO 10
300 PRINT "--------------------":RETURN
350 PRINT I;"nen me"
360 PRINT " Nendo Shoukyaku",$;S
370 PRINT "  Shoukyaku Ruikei",$;T
380 PRINT "  Zangaku",$;G
390 RETURN
```

印字例

```
< GENKA SHOUKYAKU >
--------------------
Shutoku Kagaku
    ¥ 1200000
Zanzon Kagaku
    ¥ 90000
Taiyou nensuu 7nen
--------------------
* TEIGAKU HOU *
 1nen me
  Nendo Shoukyaku
    ¥ 158571
   Shoukyaku Ruikei
    ¥ 158571
   Zangaku
    ¥ 1041429
 2nen me
  Nendo Shoukyaku
    ¥ 158571
   Shoukyaku Ruikei
    ¥ 317142
   Zangaku
    ¥ 882858
```

合計　624ステップ

メモリー内容	
G	残額
I	ループ
N	耐用年数
R	償却率
S	償却額
T	償却額累計
X	取得価額
Z	残存価額
$	キー入力

移動平均

売り上げ予測

●統計コーナー●

解説

コンピュータの「NANDEMO」社のワンマン社長が平社員の平野氏を呼んでいます．緊張して立っている平野氏へ社長が言いました．

「当社も創業以来順調に売り上げを伸ばし，3年目の決算を迎えた．そこでだ，これまでの売り上げ高から来期の売り上げ高をはじき出してくれ．

これが，これまでの当社の売り上げ高だよ．」

そう言って社長が渡したのは過去3年分の売り上げ高を書いた紙が1枚．

売り上げ高の書かれた紙を受け取って席へ戻った平野氏は，売り上げ予測をどうやればいいのかわからずためいきをつくばかりです．

見かねた美人の美子さんがそっと声を掛けてくれました．

「あのね，売り上げ高をこのPB-300へ入力するのよ．そして，移動数は5ヵ月ぐらいが普通ね．そうすれば来期の売り上げ予測ができるのよ．ま，やってみることね．」

そこで平野氏は，さっそくPB-300に載っていた「売上げ予測」のプログラムを使うことにした．

昭和57年6月から12月までの売上げが，それぞれ，

6月	403万円	7月	527万円	8月	468万円	9月	320万円
10月	345万円	11月	303万円	12月	437万円		

となっていた．そこで，プログラムを書き込んでランさせてみると，「1←I，2←C，3←O」と出た．データをインプットしなければならないから，1を押して，7個のデータを入力した．次に「Idou suu ?」と聞いてきた．移動数とは，移動平均を何ヵ月分のデータで出すのか，その月数のことです．

たとえば，5ヵ月とすれば，6～10月でIdou 1＝413，7～11月でIdou 2＝393，8～12月でIdou 3＝375が求められます．これは，5ヵ月分のデータの単純平均です．そして，同時に予測値と誤差が出ました．予測値をみると，将来的に売上げの面で非常に厳しい値が出ているのでおどろく平野氏でありました．

と，まあここまでは架空の話として，このように売上げ予測は簡単に求められます．

プログラム

プログラム入力前の設定

[MODE] [1] CLEAR [EXE]

[S] [DEFM / V] 79 [EXE]

```
LIST
 10 PRINT "*Idou He
    ikin*"
 20 INPUT "1←I,2←C,
    3←O",H$
 30 IF H$="1" THEN
    100
 40 IF H$="2" THEN
    150
 50 IF H$="3" THEN
    300
 60 GOTO 10
100 PRINT "1:Data I
    nput"
110 FOR I=1 TO 96:I
    (I)=0:NEXT I:A=
    1
120 PRINT "No.";A;"
    :";:INPUT B:IF
    B=-999;A=A-1:GO
    TO 150
130 I(A)=B:IF A<48;
    A=A+1:GOTO 120
150 PRINT "2:Idou K
    eisan","In Data
    :";A;"ken"
160 INPUT "Idou suu
    =",B:B=INT B:IF
     B≦1 THEN 160
170 C=A-B:IF B/2≠IN
    T (B/2);C=A-B+1
    :GOTO 220
200 FOR D=1 TO C:F=
    D+B:G=0:FOR E=D
     TO F:G=G+I(E):
    NEXT E
210 G=G-0.5*(I(D)+I
    (F)):I(D+48)=G/
    B:NEXT D:GOTO 3
    00
220 FOR D=1 TO C:F=
    D+B-1:G=0:FOR E
    =D TO F:G=G+I(E
    ):NEXT E
230 I(D+48)=G/B:NEX
    T D
300 PRINT "3:DISP &
     PRT"
310 INPUT "PRTer→Y/
    N",H$:IF H$="Y"
    ;MODE 7
320 PRINT "*Idou He
    ikin*":GOSUB 55
    0
330 FOR D=1 TO A:PR
    INT "In";D;"=";
    I(D):NEXT D:GOS
    UB 550
340 SET F0:FOR D=1
    TO C:PRINT "Ido
    u";D;"=";I(D+48
    ):NEXT D:SET N
400 GOSUB 550:B=0:E
    =0:F=0:G=0:FOR
    D=1 TO C
410 B=B+D*D:E=E+D:F
    =F+D*I(D+48):G=
    G+I(D+48):NEXT
    D
420 B=(F-E*G/C)/(B-
    E*E/C):F=G/C-B*
    E/C
430 SET F3:PRINT "S
    hiki","y=(";F;"
    )+(";B;")Xx":SE
    T F0
440 FOR D=1 TO C:GO
    SUB 500:G=I(D+4
    8)-I:GOSUB 510:
    NEXT D
450 GOSUB 550:FOR D
    =C+1 TO C+12:G=
    0:GOSUB 500:GOS
    UB 510:NEXT D
460 MODE 8:SET N:GO
    TO 20
500 I=F+B*D:RETURN
510 PRINT "No.";D,"
    Yosoku=";I,"  g
    osa=";G:RETURN
550 IF H$="Y";PRINT
     "--------------
    -------"
560 RETURN
```

プログラム	899ステップ
拡張メモリー	632ステップ
合　計	1531ステップ

メモリー	内容	メモリー	内容	メモリー	内容
A	データの件数	E	ループ，Σx	I	ループ，$y=a+bx$のy
B	入力データ．移動数 $y=a+bx$．Σx^2	F	移動データ $y=a+bx$のa，Σxy	I(1)〜I(48)	入力データ
		G	ΣDi，Σy，誤差	I(49)〜I(96)	移動平均値
C	移動平均データ数	H$	キー入力，JOB指定，プリンタ使用		
D	ループ				

操作

プログラムは，1がデータの入力，2が計算，3が表示・印字です．1回入力したデータで移動数を変えて再計算できます．最初の表示が「1←I，2←C，3←O」と出るので，実行する項目を1〜3で入力します．

1を入力するとデータの入力です．データの終了は−999を入力してください．続けて計算と表示を実行します．データは48件まで入力できます．

2を入力すると計算です．入力されたデータの件数を表示した後，移動数を入力します．入力したデータは配列I(1)からI(48)，移動平均の計算結果はI(49)からI(96)に記憶されます．

3を入力すると，計算した結果を受けて表示部またはプリンタへ出力します．

	手順	キー操作	表示
		S 0	＊Idou Heikin＊
メニュー表示→	1	EXE	1←I, 2←C, 3←O?
データ入力表示→	2	(例) 1 EXE	1: Data Input
	3	EXE	No 1: ?
	4	以下データを入力して繰り返す	
データ入力終了→	5	−999 EXE	Idou Keisan
計算表示→	6	EXE	In Data : 39 ken
移動数の入力→	7	EXE	Idou suu=?
表示かプリントか→	8	(例) 5 EXE	5: DISP & PRT
	9	EXE	PRTer→Y/N?
	10	(例) Y EXE	(印字される．印字例参照) 印字終了後はメニュー表示へ
		(例) N EXE	(ディスプレイ表示)

印字例

```
*Idou Heikin*
--------------------
In 1= 403
In 2= 527
In 3= 468
In 4= 320
In 5= 345
In 6= 303
In 7= 437
--------------------
Idou 1= 413
Idou 2= 393
Idou 3= 375
--------------------
Shiki
y=( 431.267)+(-19.00
0)Xx
No. 1
Yosoku= 412
  gosa= 0
No. 2
Yosoku= 393
  gosa=-1
No. 3
Yosoku= 374
  gosa= 0
--------------------
No. 4
Yosoku= 355
  gosa= 0
No. 5
Yosoku= 336
  gosa= 0
No. 6
Yosoku= 317
  gosa= 0
No. 7
Yosoku= 298
  gosa= 0
No. 8
Yosoku= 279
  gosa= 0
No. 9
Yosoku= 260
  gosa= 0
No. 10
Yosoku= 241
  gosa= 0
No. 11
Yosoku= 222
  gosa= 0
No. 12
Yosoku= 203
  gosa= 0
No. 13
Yosoku= 184
  gosa= 0
No. 14
Yosoku= 165
  gosa= 0
No. 15
Yosoku= 146
  gosa= 0
```

年	月	売り上げ In	移動平均 Idou	予測値 Yoso Ku	誤差 gosa
57					
	6	403			
	7	527			
	8	468	413	412	0
	9	320	393	393	−1
	10	345	375	374	0
	11	303	今期	355	0
	12	437	今期	336	0
58					
	1		来期の売り上げ予測	317	0
	2			298	0
	3			279	0
	4			260	0
	5			241	0
	6			222	0
	7			203	0
	8			184	0
	9			165	0
	10			146	0

移動数　5項

回帰直線の式　$y=(431.267)+(-19.000)\times x$

出力は，入力したデータ，移動平均の計算結果，直線回帰の式，回帰式の計算結果（予測値）と誤差です．予測値は，直線回帰式のxに1から順に代入して計算した結果です．移動平均のデータと同数までは，移動平均値−予測値で誤差を出力します．移動平均のデータ数以上では予測値のみ12項まで計算します．このとき誤差は0です．

誤差0で出力される予測値のうち前から（移動数÷2の小数部切り捨て）2個は今期の分です．これは，移動平均法の性質上発生するものです．

順位相関係数

アンケート分析

●統計コーナー●

あなたとあなたの恋人の相性はどの程度？１が出れば相性はピッタリ！－１に近ければ考えもの．そんな利用法も楽しめるプログラムです．



解説

商品の展示会を開催し，何点かの主力商品を展示します．来場者にはアンケート用紙を配布して，いいなと思うものの順と買いたいなと思うものの順に１番から順位をつけてもらいます．

アンケートの結果を集計し，度数分布表に描いてみたけれど両者の傾向がはっきりつかめません．展示方法や商品構成に問題はなかったかを知りたいし，人気の商品と実売に結びつく商品の関係も知りたいのです．

また国語の成績順と英語の成績順を見比べて首をひねっている先生がいます．こちらは，国語に強い子の英語の成績はどうかを知りたがっているようです．

■　順位相関係数を使って分析すれば，たちまち関係がわかる！

どちらも順位を係数としてとらえることができたら解決しそうです．パソコンへ比較する２つの順位を入力してみましょう．結果が出ました．

初めに「順位相関係数」と出力されます．これが係数の名まえです．次は，スピアマンという係数名とその値が出力されます．もう一つ別の計算方法で計算したケンドールという係数名とその値も出力されます．

順位相関係数には２つの計算方法があるので両方を計算しています．

次は，入力したデータのNo.と順が表に出力されます．

■　結果は１に近いほど関係が強い

この結果をもとに分析をすすめます．係数は－１から＋１の値を示します．

１に近い値を示せば，１に近い程２つの関係は強いことを意味します．展示方法や商品構成が来場者の好みに一致したと言えます．また，国語に強ければ英語にも強いことも言えます．

－１や＋１は順位がまったく逆か，同じでなければ得られない値です．

例題の順位相関係数は，0.558と0.373です．例えば，－１を０％とし，０を50％，１を100％とすれば，係数0.558の場合，77.5％の相関がある，ということになります．0.373の場合は，68.5％です．このようにして，人気分野と進出希望分野の相関を求めることができるのです．

プログラムは，データのNo.１からAのつけた順位，Bのつけた順位で交互に入力します．データは40件まで入力できます．

データのNo.が40に達するか，Aの順位を聞かれたとき－999を入力するとデータの入力は終わりです．２つの順位相関係数を計算したら，プリンタの使用を問います．Ｙを入力すればプリンタへ出力します．出力は順位相関係数を入力した順位です．

操作

例として右表の順位相関係数を計算してみましょう．

プログラム実行の前に必ずDEF M 64で変数エリアの拡張をしてください．

実行すると，「Juni Soukan」と表示され，「Data END＝－999」(データ入力終了時は，－999を入力せよ）と表示されます．

次に「No. 1 Juni」に続いて「A wa:?」と表示されるので，１位のAの値（例では１）を入力し，「B wa:?」でBの値（例では７)を入力します．

この繰り返しで，データをどんどん入力していき，終了時には，「A wa:?」のとき－999を入れればプリンタを使用するかを聞いてきますから，Yを入力して終わりです．右表を入力した印字例を次ページに示します．

「上場企業の新成長分野調査」日本経済新聞 1982年12月５日より．（注）A：人気ランキング，B：実際の進出・拡大分野ランキング

分　野	A	B
バイオテクノロジー	1	7
ロボット・ＥＡ	2	4
光通信	3	13
コンピュータ利用技術	4	5
新素材	5	1
ＯＡ機器	6	2
半導体	7	8
健康産業	8	5
代替エネルギー	9	15
省エネルギー製品	10	3
コンピュータ機器	11	14
防衛機器	12	18
消費者金融	13	16
レジャー用品	14	10
音響・映像製品	15	9
原子力関連	16	11
医療機器	17	12
ニューメディア	18	16

プログラム

プログラム入力前の設定

MODE 1 CLEAR EXE

S DEFM V 64 EXE

```
10 PRINT "Juni Sou
   kan"
20 PRINT "DataEND=
   -999":H=1
30 PRINT "No.";H;"
   Juni";:GOSUB 31
   0
40 INPUT "A wa:",A
   :A=INT A:IF A=-
   999;H=H-1:GOTO
   70
50 INPUT "B wa:",B
   :B=INT B
60 J(H)=A:J(H+40)=
   B:IF H<40;H=H+1
   :GOTO 30
70 C=0:FOR I=1 TO
   H:C=C+(J(I)-J(I
   +40))↑2:NEXT I
80 D=1-6*C/(H↑3-H)
90 E=0:F=0:FOR I=1
    TO H-1:FOR J=I
   +1 TO H
100 IF SGN (J(I)-J(
   J))=SGN (J(I+40
   )-J(J+40));E=E+
   1:GOTO 120
110 F=F+1
120 NEXT J:NEXT I
130 G=(E-F)/(H*H-H)
   *2
140 INPUT "PRTer→Y/
   N",$:IF $="Y";M
   ODE 7
150 SET F3
160 PRINT "Juni Sou
   kan keisuu","--
   ---------------
   ---"
170 PRINT "SPEARMAN
   : r=";D,"KENDAL
   L : t=";G
180 SET N
190 PRINT " No!  A!
     B"
200 PRINT "---+---+
   ---"
210 FOR I=1 TO H
220 J=I:GOSUB 300:P
   RINT J;"!";
230 J=J(I):GOSUB 30
   0:PRINT J;"!";
240 J=J(I+40):GOSUB
     300:PRINT J
250 NEXT I
260 MODE 8
270 PRINT "END":END

300 IF J<10;PRINT "
    ";
305 RETURN
310 FOR I=1 TO 200:
   NEXT I:PRINT CS
   R 0;"
     ";
320 PRINT CSR 0;:RE
   TURN
```

印 字 例

```
Juni Soukan keisuu
--------------------
SPEARMAN: r= 0.558
KENDALL : t= 0.373
 No!  A!  B
---+---+---
  1!  1!  7
  2!  2!  4
  3!  3! 13
  4!  4!  5
  5!  5!  1
  6!  6!  2
  7!  7!  8
  8!  8!  5
  9!  9! 15
 10! 10!  3
 11! 11! 14
 12! 12! 18
 13! 13! 16
```

プログラム 625ステップ
拡張メモリー 512ステップ
合 計 1137ステップ

メモリ内容	
A	A側の順位入力
B	B側の順位入力
C	Σ (ai−bi)²
D	スピアマンの係数
E	同方向の順位づけ
F	異方向の順位づけ
G	ケンドールの係数
H	データ件数
I	ループ
J	ループ
J(1)〜J(40)	A側のデータ
J(41)〜J(80)	B側のデータ

母集団の推定

抜き取り検査

●統計コーナー●

穴あけ作業を1000個行ない，そのうち20個を取出して寸法を測定しました．20個のデータをもとに，1000個の穴の寸法はどの範囲か，また寸法の平均はいくらかを推定する．こんなときに利用するプログラムです．



解説

多くの集まりの中から，ランダムに何個かのサンプルを取出すと，そのサンプルの属していた集団を代表する値として用いることができ，逆に，全体がどの範囲にあるかを推定することができます．これを母集団の推定といい，取出すサンプルが多いほど全体の推定は正確に行なえます．

推定は，全体の68%，95%，99.7%の3つの範囲に分けますが，一般に使われる頻度の高い，信頼度95%の信頼区間で母集団を推定します．

入力は，母集団の総数とサンプルデータです．100個まで入力できます．出力の項目は，母集団の総数とサンプルの数，平均値，母集団総計値の実現値，信頼度95%の信頼区間での母集団平均値と母集団総計値，入力したサンプルデータを出力します．

■ 計算式

サンプル平均値 $x_n = \Sigma x_i / n$ 　　n：サンプル数

母集団総計値の実現値 $t_n = N\overline{x_n}$ 　　N：母集団総数

不偏推定量の実現値 $U_n{}^2 = \dfrac{N-1}{N} \cdot \dfrac{1}{n-1} \Sigma (x_i - \overline{x_n})^2$

信頼度95%の信頼区間

母集団平均値（μ） $\mu = x_n \pm 1.96 \times \sqrt{\dfrac{N-n}{N-1} \cdot \dfrac{U_n{}^2}{n}}$

母集団総計値（τ） $\tau = t_n \pm 1.96 \times N \times \sqrt{\dfrac{N-n}{N-1} \cdot \dfrac{U_n{}^2}{n}}$

プログラム

プログラム入力前の設定

MODE 1 CLEAR EXE

S DEFM V 86 EXE

```
LIST
  10 PRINT "*Boshuud
     an*"
  20 GOSUB 1000
  30 GOSUB 2000
  40 GOSUB 3000
  50 GOTO 10
1000 PRINT "*Data In
     put*"
1010 A=1
1020 INPUT "Sousuu:"
     ,B
1030 PRINT "Data No.
     ";A;:INPUT D:IF
      D=-999;A=A-1:G
     OTO 1050
1040 L(A)=D:IF A<100
     ;A=A+1:GOTO 103
     0
1050 RETURN
2000 PRINT "* KEISAN
      *";
2050 G=0:FOR I=1 TO
     A:G=G+L(I):NEXT
      I
2060 H=G/A:J=B*H
2070 D=0:FOR I=1 TO
     A:D=D+(L(I)-H)↑
     2:NEXT I
2080 C=(B-1)*D/(B*(A
     -1))
2090 G=1.96*SQR ((B-
     A)*C/((B-1)*A))
2100 D=H-G:E=H+G:K=J
     -G*B:L=J+G*B
2110 PRINT CSR 0;
2120 RETURN
3000 PRINT "*DISP &
     PRT*"
3010 INPUT "PRTer→Y/
     N",$:IF $="Y";M
     ODE 7
3020 $="       "
3030 PRINT "Boshuuda
     n Zensuu",$;"N:
     ";B
3040 PRINT "Sample s
     uu n:";A
3050 SET F3:PRINT "S
     ample Heikin",$
     ;"x:";H
3060 PRINT "Sample S
     oukeichi",$;"tn
     :";J
3070 PRINT $;"Un:";C
     :GOSUB 4000
3080 PRINT "Sinraido
      95% no"
3090 PRINT "*Heikinc
     hi Kukan μ","("
     ;D;",";E;")"
3100 PRINT "*Soukeic
     hi Kukan τ","("
     ;K;",";L;")"
3105 GOSUB 4000:SET
     N
3110 PRINT "SAMPLE D
     ata"," No.  Dat
     a"
3120 FOR I=1 TO A:IF
      I<10;PRINT " "
     ;
3130 PRINT I;"  ";L(
     I):NEXT I:MODE
     8
3140 INPUT "PRT END→
     Y/N",$:IF $="N"
      THEN 3010
3150 RETURN
4000 PRINT "--------
     ------------":R
     ETURN
```

プログラム　777ステップ
拡張メモリー　688ステップ
合計　1465ステップ

メモリー内容	
A	データ件数（n）
B.	母集団総数〔N〕
C	$U_n{}^2$
D	データ入力，母集団平均値1（下）
E	母集団平均値μ(上)
G	Σx_i，計算
H	$\overline{x_n}$
I	ループ
J	t_n
K	τ(下)
L	τ(上)
L(1)〜L(100)	データ配列
$	キー入力

操作

プログラムの入力前か実行前に，必ずDEFM 86で変数エリアを拡張してください。

	手順	キー操作	表示
		MODE 0	READY P0
	1	S 0	＊Boshuudan＊
	2	EXE	＊Data Input＊
母集団総数入力	3	EXE	Sousuu：?（母集団総数の入力）
	4	（例）980 EXE	Data No.1?（サンプルデータ）
サンプルデータを次々と入力→	5	（例）68.5 EXE	Data No.2?（サンプルデータの数だけ繰り返し）
データ終了は−999を入力→	6	−999 EXE	（計算中の表示）＊KEISAN＊
	7	EXE	＊DISP & PRT＊
プリンタ使用？→	8	EXE	PRTer→Y/N?
	9	（例）Y EXE	（プリント印字）
プリント終了か？→	10		PRT END→Y/N?
	11	（例）Y EXE	（表示1に戻る）

印字例

```
Boshuudan Zensuu
    N: 980
Sample suu n: 30
Sample Heikin
    x: 63.597
Sample Soukeichi
    tn: 62324.733
    Un: 30.341
--------------------
Sinraido 95% no
*Heikinchi Kukan μ
( 61.655, 65.538)
*Soukeichi Kukan τ
( 60421.871, 64227.5
96)
--------------------
```

```
SAMPLE Data
 No.  Data
  1   68.5
  2   69.4
  3   70.1
  4   58.1
  5   64.7
  6   54.2
  7   55.3
  8   70.6
  9   64.3
 10   62.1
 11   62.2
 12   68.5
 13   70.6
 14   58.2
 15   55
 16   49.9
 17   60
 18   61.3
 19   58.3
 20   65.7
 21   65.9
 22   68.4
 23   68.8
 24   66.2
 25   64.6
 26   69.6
 27   61.3
 28   66.3
 29   67
 30   62.8
```

度数分布表（品質管理や成績処理に）

●グラフコーナー●

度数というのは，同じ値あるいは同じクラスの値が出現する回数で，度数分布表というのは各クラスのデータ出現回数を表にまとめたものです．非常に応用範囲の広い，統計処理の基本的なプログラムです．



解説

項目名，データの最大値，最小値，クラスの数を入力した後，データを1件ずつ入力します．入力したデータが最大値と最小値の範囲に入っていなければ，誤入力を表示してカウントしません．

データが終了すれば，−999を入力すると計算を行ないます．計算が終われば出力です．プリンタ使用の有無を問合わせてきます．プリンタへ印字するには[Y]を押してください．

出力する項目は，項目名，データの最大値，最小値，データ件数，平均，平方和，分散，標準偏差，クラスの数，クラスの幅，表の「＊」1個の値をまず表示し，次に度数分布表，各クラスの範囲と数です．盛り沢山の内容なのでプリンタへ印字した方がわかりやすいでしょう（印字例参照）．

入力したデータは個々に記憶されませんので注意してください．配列は各クラスの個数をカウントするためのものです．

クラスの幅は，データの最大値，最小値を含むようにプログラム内で計算しています．度数分布表の「＊」1個の値も，クラスの最大個数に合わせてプログラム内で設定しています．変数エリアの拡張は，DEFM 10 です．

プログラム

プログラム入力前の設定

[MODE] [1] CLEAR [EXE]

[S] [DEFM V] 10 [EXE]

```
5 VAC
10 PRINT "Dosuu-Bunpu"
15 $="ABCDEFGHIJ"
20 INPUT "Koumokumei:",A$
30 INPUT "DATAmax.=",L
40 INPUT "DATAmin.=",S
50 INPUT "Class no kazu(2-10):",K:K=INT (ABS K)
60 IF K>10 THEN 50
70 H=(L-S)/K+(L-S)/(L*K):G=S-(L-S)/(2*L)
80 N=0:O=0:P=0
90 PRINT CSR 0;"No.";N+1;":";:INPUT X
100 IF X=-999 THEN 180
110 IF X>L;PRINT "Data Max OVER":GOTO 90
120 IF X<S;PRINT "Data UNDER Min":GOTO 90
130 B=INT ((X-G)/H):Z(B)=Z(B)+1
140 N=N+1:O=O+X*X:P=P+X
150 GOTO 90
180 M=P/N:Q=O-P*P/N:R=Q/N:T=SQR R
190 C=0
200 FOR I=0 TO K-1:IF C<Z(I);C=Z(I)
210 NEXT I
220 D=18/C
230 INPUT "PRTer→Y/N",J$:IF J$="Y";MODE 7
240 PRINT "[ ";A$;" ]":GOSUB 500
250 PRINT "DATA","max.:";L,"min.:";S,"suu :";N
260 SET F3:PRINT "heikin :";M,"heihouwa:";Q,"bunsan :";R
270 PRINT "hyoujunhensa","  :";T:SET N
280 GOSUB 500
290 PRINT "CLASS"," suu :";K
300 SET F3:PRINT " Range:";H:GOSUB 500
310 PRINT "Dosuu bunpu","   *=";1/D
330 PRINT " 0_________________"
340 FOR I=1 TO K
350 PRINT MID(I,1);"!";
360 E=INT (Z(I-1)*D)
370 IF E=0 THEN 390
380 PRINT "*";:E=E-1:GOTO 370
390 PRINT ""
400 NEXT I
410 PRINT ""
420 PRINT "Class,Range,Count"
430 FOR I=1 TO K:GOSUB 500
432 IF I=1;IF G<0;F=0:GOTO 440
434 F=G+H*(I-1)
440 SET F3:PRINT MID(I,1);":",F;"to";G+H*I
445 SET N:PRINT " Count:";Z(I-1)
450 NEXT I
460 SET N:MODE 8
470 PRINT "END"
480 END
500 IF J$="Y";PRINT "--------------------"
510 RETURN
```

プログラム	978ステップ
拡張メモリー	80ステップ
合計	1058ステップ

メモリー内容		H	クラスの幅	Q	偏差平方和
A$	項目名（入力）	I	ループ	R	分散
B	クラス分け	J$	プリンタ使用入力	S	データの最小値
C	度数の最大値	K	クラスの数（入力）	T	標準偏差
D	度数分布表スケーリング	L	データの最大値	X	入力したデータ
E	印字する＊個数	M	データ平均 $\overline{x}$	Z(1)	
F	各クラスの下限	N	データの件数	～	各クラス度数
G	クラスの初期値	O	Σx^2	Z(10)	
		P	Σx		

操作は比較的簡単で，表示の問うとおりに入力していってください．ここでは，算数のテスト成績（100点満点，81人）を入力した結果を印字してみました．

印字例

```
[ SANSUU ]
------------------
DATA
max.: 100
min.: 0
suu : 81
heikin  : 58.914
heihouwa: 27730.395
bunsan  : 342.351
hyoujun hensa
        : 18.503
------------------
CLASS
 suu  : 10
 Range: 10.100
------------------
Dosuu bunpu
   *= 1.333
 0_______________
A!
B!*
C!**
D!***
E!********
F!************
G!******************
H!*******
I!***
J!**
```

```
Class,Range,Count
------------------
A:
 0.000 to 9.600
 Count: 1
------------------
B:
 9.600 to 19.700
 Count: 2
------------------
C:
 19.700 to 29.800
 Count: 3
------------------
D:
 29.800 to 39.900
 Count: 5
------------------
E:
 39.900 to 50.000
 Count: 11
------------------
F:
 50.000 to 60.100
 Count: 17
------------------
G:
 60.100 to 70.200
 Count: 24
```

```
------------------
H:
 70.200 to 80.300
 Count: 10
------------------
I:
 80.300 to 90.400
 Count: 5
------------------
J:
 90.400 to 100.500
 Count: 3
```


万能棒グラフ

●グラフコーナー●

解説

7件までのデータを棒グラフに表わします．入力できるのは，グラフの名称，データの件数，各データの名称と値，Y軸の単位です．Y軸の単位は文字変数ですから，7文字までなら，¥，$，km，kg·cm/s，UNIT ……など何でも入力できます．

グラフのスケールは，データの最大値から計算し，プログラムで自動的にスケーリング（目盛づけ）をします．

出力は，棒グラフ，「■」1個あたりの表示数量，単位，各データの名称と値および比率です．

あらゆる用途を考えてプログラムしてあります から，いろいろ工夫して印字してみてください．

★ワンポイントアドバイス★

この万能棒グラフは，横の目盛がない方がきれいで見やすいと思いますが，横ケイの目盛を入れたい人は，

3080行の PRINT "　■" を
　　　PRINT "─■" に，

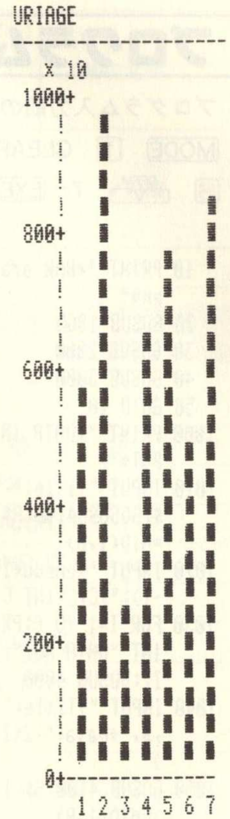
同様に，

3090行の PRINT "　　" を
　　　PRINT "──" に，

3100行の PRINT " " を，
　　　PRINT "─" に，

してください．横のケイが入ります．

印字例




```
------------------
■= 40
Y tani:¥ x 10
------------------
Data No. 1  BEER
   5800  12.6%
Data No. 2  HOT DOG
   9700  21.1%
Data No. 3  MILK
   2650  5.8%
Data No. 4  POPCORN
   6780  14.7%
Data No. 5  JUICE
   7820  17.0%
Data No. 6  CARAMEL
   4900  10.6%
Data No. 7  WHISKY
   8430  18.3%
```

プログラム

プログラム入力前の設定

[MODE] [1] CLEAR [EXE]

[S] [DEFM/V] 7 [EXE]

```
  10 PRINT "*BAR gra
     ph*"
  20 GOSUB 1000
  30 GOSUB 2000
  40 GOSUB 3000
  50 GOTO 10
1000 PRINT "*DATA IN
     PUT*"
1010 INPUT "Title:",
     $:GOSUB 4100:A$
     =MID(1,B)
1020 INPUT "Kensuu(1
     -7):",C:C=INT C
1030 FOR I=1 TO C:PR
     INT "DATA No.";
     I;:GOSUB 4000
1040 INPUT "→Title:"
     ,$,"→Data:",Z(I
     )
1050 GOSUB 4100:S$(I
     )=MID(1,B)
1060 NEXT I
1070 INPUT "DATA OK→
     Y/N",$:IF $="N"
      THEN 1020
1080 INPUT "Y tani:"
     ,$:GOSUB 4100:D
     $=MID(1,B)
1090 RETURN
2000 PRINT "*KEISAN*
     ";
2010 E=0:F=Z(1)
2020 FOR I=1 TO C:E=
     E+Z(I):IF F<Z(I
     );F=Z(I)
2030 NEXT I
2040 G=INT (LOG F):I
     F G≦0;B=LOG (10
     ↑G/100):GOTO 20
     50
2042 IF G≦2;B=0
2044 IF G>2;B=G-2
2050 H=(INT (F/10↑G)
     +1)*10↑(G-B):F=
     H/25
2060 PRINT CSR 0;
2070 RETURN
3000 PRINT "DISP & P
     RT"
3010 INPUT "PRTer→Y/
     N",$:IF $="Y";M
     ODE 7
3020 PRINT A$:GOSUB
     4200:K=25:IF B≠
     0;PRINT "   x";
     10↑B
3030 FOR I=H TO F ST
     EP -F
3040 IF K/5≠INT (K/5
     );PRINT "    !"
     ;:GOTO 3080
3050 IF I<100;PRINT
     " ";
3060 IF I<10;PRINT "
      ";
3070 PRINT I;"+";
3080 FOR J=1 TO C:IF
      Z(J)/10↑G*10↑(
     G-B)≧I;PRINT "
     ■";:GOTO 3100
3090 PRINT "  ";
3100 NEXT J:K=K-1:PR
     INT ""
3110 NEXT I
3120 PRINT "   0+---
     -----------"
3130 PRINT "     ";:
     FOR I=1 TO C:PR
     INT I;:NEXT I:P
     RINT ""
3140 GOSUB 4200:PRIN
     T " ■=";F,"Y ta
     ni:";D$;
3150 IF B≠0;PRINT "
     x";10↑B;
3160 PRINT "":GOSUB
     4200
3170 FOR I=1 TO C:PR
     INT "Data No.";
     I;"   ";S$(I)
3180 PRINT "   ";Z(I
     );" ";:SET F1:P
     RINT Z(I)/E*100
     ;"%"
3190 SET N:NEXT I
3200 MODE 8:INPUT "E
     ND→Y/N",$:IF $=
     "N" THEN 3010
3210 RETURN
4000 FOR J=1 TO 150:
     NEXT J:PRINT CS
     R 0;"
       ";:PRINT CSR
     0;
4010 RETURN
4100 B=LEN($):IF B>7
     ;B=7
4110 RETURN
4200 PRINT "--------
     ------------":R
     ETURN
```

プログラム	995ステップ
拡張メモリー	56ステップ
合計	1051ステップ

メモリー内容					
		F	データ最大値，1目盛の値	S$(1)～S$(7)	X軸の名称
A$	グラフの名称				
B	文字列の長さ，Y軸倍率	G	データ最大値桁数		
		H	目盛の最大値	Z(1)～Z(7)	データの値
C	データの件数	I	ループ		
D$	Y軸単位	J	ループ		
E	Σx_i	K	ループ逆カウンタ	$	キー入力

操　作

プログラム実行の前にDEFM 7を確認してください．

	手順	キー操作	表　示
		[MODE] [0]	READY P0
見出し表示→	1	[S] [0]	＊BAR GRAPH＊
	2	[EXE]	＊DATA INPUT＊
	3	[EXE]	Title : ?
タイトルを入力→	4	(例)URIAGE [EXE]	Kensuu (1−7) ?
件数を入力→	5	(例) 7 [EXE]	DATA No. 1 →Title : ?
1番目の名称入力→	6	(例) BEER [EXE]	→Data : ?
1番目のデータ入力→	7	(例) 5800 [EXE]	→Title : ? (データ入力の繰り返し) 印字例に従って入力して下さい
データの確認→			DATA OK→Y/N? (Nを入力すれば表示4へ)
		(例) Y [EXE]	Y tani : ?
Y軸の単位入力→		(例) ¥	＊KEISAN＊
			DISP & PRT
プリンタ使用？→		[EXE]	PRTer →Y/N?
		(例) Y [EXE]	(プリンタに印字)
Nなら再度，表示印字へ戻る．Yなら手順1へ戻る→		(例) Y [EXE]	END→ Y/N? ＊BAR graph＊

基準化と偏差値グラフ

●グラフコーナー●

解説

データが集団の中でどこに位置するかを見るのに，基準化，標準化が使われます．ここでは，データを入力することで，データの平均，平方和，分散，標準偏差を計算し，各データごとに基準化した値と偏差値を表示，または印字します．

プリンタへ出力するなら，基準化データのグラフを描きます．入力したデータに誤りがあれば，訂正または削除できます．

■ 計算式

平均	$\bar{x}=\Sigma x_i/N$
平方和	$S=\Sigma x_i^2-(\Sigma x_i)^2/n$
分散	$s^2=S/n$
標準偏差	$s=\sqrt{s^2}$
基準化	$Z_i=(x_i-\bar{x})/S$
偏差値	$y_i=aZ_i+b=10Z_i+50$

データは59件まで入力できます．データの件数を増したい場合は，500行から620行の訂正ルーチンを削除し，130行の59を拡張したデータ数に合わせて変更してください．データの入力終了は，59件入力するか－999を入力したときです．なおプログラム実行時には，必ずDEFM 47で変数エリアを拡張してください．

メモリー内容		F	平方和 S	M	訂正時のデータNo.
		G	分散 s^2	N	データ件数
A$	タイトル	H	標準偏差 s		
B	データの入力	I	ループ		
C	Σx_i				
D	Σx_i^2				
E	平均 $\bar{x}$	K	偏差値		
		L$	キー入力		

プログラム

プログラム入力前の設定

MODE 1 CLEAR EXE

S DEFM 47 EXE

```
10 PRINT "* Hensachi *"
20 INPUT "1←I,2←O,3←T",L$
30 IF L$="1" THEN 100
40 IF L$="2" THEN 150
50 IF L$="3" THEN 500
60 GOTO 10
100 PRINT "1:Data Input";:GOSUB 700
110 N=1:INPUT "Title:",A$
120 PRINT "Data";N;":";:INPUT B:IF B=-999;N=N-1:GOTO 150
130 N(N)=B:IF N<59;N=N+1:GOTO 120
150 C=0:D=0:FOR I=1 TO N:C=C+N(I):D=D+N(I)*N(I):NEXT I
160 E=C/N:F=D-C*C/N:G=F/N:H=SQR G
200 PRINT "2:DISP & PRT";:GOSUB 700
220 INPUT "PRTer→Y/N",L$:IF L$="Y";MODE 7
230 PRINT A$:GOSUB 750:PRINT "Data:";N;"ken"
240 SET F3:PRINT "heikin  :";E,"heihouwa:";F,"bunsan  :";G
250 PRINT "hyoujun hensa","  :";H:GOSUB 750:SET N
260 FOR I=1 TO N:J=(N(I)-E)/H:K=INT (10*J+50)
270 SET N:PRINT "No.";:IF I<10;PRINT " ";
280 PRINT I;"  in  :";N(I):SET F3:PRINT "    kijunka :";J
290 SET N:PRINT "   hensachi:";K
300 NEXT I:GOSUB 750:IF L$≠"Y" THEN 430
310 $="        !        "
320 PRINT "Kijunka graph","-2 -1  0  1  2"
330 PRINT "+---+---+---+---+No."
340 FOR I=1 TO N:J=INT (((N(I)-E)/H+2)*17/4):IF J<0;J=0
350 IF J>16;J=16
360 IF J=0;PRINT "*";:GOTO 390
370 PRINT MID(1,J);"*";
380 IF J=16 THEN 400
390 PRINT MID(J+2,16-J);
400 IF I<10;PRINT " ";
410 PRINT I
420 NEXT I:MODE 8
430 GOTO 20
500 PRINT CSR 0;"3:Teisei";:GOSUB 700
510 INPUT "DataPRT→Y/N",L$:IF L$≠"Y" THEN 580
530 INPUT "PRTer→Y/N",L$:IF L$="Y";MODE 7
550 GOSUB 750:PRINT "Input Data":FOR I=1 TO N:IF I<10;PRINT " ";
560 PRINT I;":";N(I)
570 NEXT I:MODE 8
580 INPUT "Teisei→Y/N",L$:IF L$≠"Y" THEN 20
600 INPUT "Data No.:",M:M=INT (M):PRINT "Data:";N(M)
610 INPUT "Teisei Data",B:IF B≠-999;N(M)=B:GOTO 580
620 FOR I=M TO N:N(I)=N(I+1):NEXT I:N=N-1:GOTO 580
700 FOR I=1 TO 200:NEXT I
710 PRINT CSR 0;"                ";:PRINT CSR 0;:RETURN
750 IF L$="Y";PRINT "--------------------"
760 RETURN
```

プログラム	1192ステップ
拡張メモリー	376ステップ
合計	1568ステップ

操作

プログラム実行前に必ずDEFM 47 を確認してください.

	手順	キー操作	表　　示
1はデータ入力, 2は計算と出力, 3は訂正を示す→		MODE 0	READY P0
	1	S 0	＊　Hensachi　＊
	2	EXE	1←I, 2←O, 3←T?
タイトル入力→	3	(例) 1　EXE	1 : Data Input Title : ?
	4	(例) KOKUGO EXE	Data 1 : ?
データを次々と入力する→	5	(例) 53　EXE	Data 2 : ? (以下データ入力繰り返し)
データ終了は−999を入力→	6	−999　EXE	2 : DISP & PRT PRTer→Y/N?
プリンタ使用？→	7	(例) Y　EXE	(プリンタで印字・印字例参照)
手順3で3を入力→	9		3 : Teisei Data PRT→Y/N?
	10	(例) N　EXE	Teisei→Y/N?
	11	(例) Y　EXE	Data No.: ?
	12	(例) 20　EXE	Data 68
訂正データを入力, 削除するときは−999を入力→	13	EXE	Teisei Data?
	14	65　EXE	Teisei→Y/N?

印 字 例

```
RIKA
--------------------
Data: 33ken
heikin  : 59.667
heihouwa: 10563.333
bunsan  : 320.101
hyoujun hensa
        : 17.891
--------------------
No.  1  in   : 54
     kijunka:-0.317
    hensachi: 46
```

```
No.  2  in   : 68
     kijunka: 0.466
    hensachi: 54
No.  3  in   : 59
     kijunka:-0.037
    hensachi: 49
No.  4  in   : 78
     kijunka: 1.025
    hensachi: 60
No.  5  in   : 50
     kijunka:-0.540
    hensachi: 44
```

省略

```
     kijunka:-0.093
    hensachi: 49
--------------------
Kijunka graph
-2 -1   0   1   2
+---+---+---+---+No.
       *!          1
        ! *        2
        *          3
        !   *      4
      * !          5
        !*         6
   *    !          7
*       !          8
        !      *   9
```

省略

第4編

巻末参考編

PB－300命令一覧表

1．演算記号

(1)算術演算

名　称	一般の使い方	使い方	意　味	計算の優先順
べき乗	x^y	X↑Y	XのY乗	1
負　号	$-x$	－X	マイナスX	2
かけ算	$x\times y$	X＊Y	XとYを掛ける	3
わり算	$x\div y$	X／Y	XをYで割る	3
たし算	$x+y$	X＋Y	XとYを足す	4
ひき算	$x-y$	X－Y	XからYを引く	4
代　入	$x=y+5$	X＝Y＋5	XにY＋5の値を代入	5

(2)関係演算（条件式）

一般の使い方	使い方	意　味
$x=y$	X＝Y	XとYが等しい
$x\neq y$	X≠Y	XとYは等しくない
$x<y$	X＜Y	XはYより小さい
$x>y$	X＞Y	XはYより大きい
$x\leqq y$	X≦Y	XはYより小さいか等しい
$x\geqq y$	X≧Y	XはYより大きいか等しい

（数値どうしであれば，定数，変数，式．
文字どうしは定数と変数が比較できます）

2．特殊文字

一般の使い方	使い方	意　味
$x\times 10^y$	XＥY	指数部置数，10のY乗
π	π	円周率　3．141592654

3．組み込み関数

名　称	一般の使い方	コマンド使用例	意味，注意
三角関数	sin x cos x tan x	SIN X COS X TAN X	
逆三角関数	$\sin^{-1} x$ $\cos^{-1} x$ $\tan^{-1} x$	ASN X ACS X ATN X	
対数関数	log x ln x	LOG X LN X	$\log_{10}$X（常用対数） $\log_e$ X（自然対数）
指数関数	e x^y	EXP 1 X↑Y	自然対数の底 e XのY乗計算
平方根	$\sqrt{x}$	SQR X	
符号化		SGN X	X＜0なら－1 X＝0なら　0　を得る X＞0なら　1
乱数		RAN＃	0＜RAN＃＜1の疑似乱数を発生
四捨五入		RND（X, Y）	Xの値を10^Yの桁で四捨五入
絶対値化	$\|x\|$	ABS X	Xの絶対値を得る
整数化		INT X	X＞0なら小数点以下切り捨て X＜0なら|X|の小数点以下を切り上げた値に－がつく INT（ 1.2）→1 INT（－1.2）→－2
整数部除去		FRAC X	小数点以下の数値のみ取り出す．絶対値表示ではないので±の符号は付く．

4．文字関数

用　途	コマンド	使用例	使用例意味
文字変数の文字数を求める	LEN	K＝LEN（A$）	文字変数A$に記憶されている文字の数をKに代入
文字列の数字を数値に変換	VAL	C＝VAL（J$）	文字変数J$に記憶されている数字のみの文字列を数値に変換してCに代入
文字を指定した数だけ取り出す	MID	N$＝MID（9, 5）	専用文字変数$の文字列の先頭から9番目より5個を取り出しN$に代入

5.マニュアル・コマンド

用　途	コマンド	使用例	使用例意味
プログラムの消去	CLEAR	CLEAR	現在指定されているプログラムエリアのプログラムを消去
	CLEAR A	CLEAR A	プログラムエリアP 0からP 9に記憶されているすべてのプログラムを消去
変数エリアの大きさ指定	DEFM	DEFM	現在指定されている変数エリアの大きさを表示
		DEFM 0	拡張されていた変数エリアを基本の26に戻す
		DEFM 20	引数は1から196まで．指定した数だけ変数エリアを拡張
プログラムのリスト	LIST	LIST	現在指定されているプログラムエリアのプログラムを行番号の小さい方から表示
		LIST 100	行番号100から表示
	LIST A	LIST A	プログラムエリアP 0からP 9まで順に記憶されているすべてのプログラムを表示
プログラムの実行開始	RUN	RUN	現在指定されているプログラムエリアのプログラムで行番号の小さい方から実行を開始
		RUN 100	行番号100からプログラムの実行を開始
	[S] 0~9	[S] 2	プログラムエリアP 2のプログラムを行番号の小さい方から実行開始

DEFM，LISTとLIST Aは，コマンドを実行する前に[MODE] 7(プリンタの使用)を指定することで，結果をプリンタへ印字します．
表示とプリンタへの印字は目的により使い分けます．

6.プログラム・コマンド，関数

用　途	コマンド	使用例	使用例意味
プログラムの実行終了	END	END	END を実行するとプログラムの実行を終了する．
ループ(繰り返し)	FOR TO STEP ⁓ NEXT	FOR I=5 TO 21 STEP 0.5 ⁓ NEXT I	FOR　変数名=初期値　TO　終値　STEP　増分． FORとNEXTに囲まれた処理を，変数Iが5から始まり1回に0.5ずつ足して21かまたは越えない範囲で，繰り返し実行する． STEP 増分は省略可．
サブルーチン	GOSUB	GOSUB 500	サブルーチンを500行から実行する．
		GOSUB #2	サブルーチンとしてプログラムエリアP 2を実行する．
			サブルーチンの終わりRETURNを実行すると，GOSUBの次に書かれた命令へ戻り実行を始める．
無条件分岐	GOTO	GOTO 100	100行にプログラムの実行を移す．
		GOTO #2	プログラムエリアP 2にプログラムの実行を移す．
条件分岐	IF 条件式 THEN	IF I>9 THEN 100	条件式が成立すれば THEN の後に書かれた行番号へ実行を移す． Iが9より大きいなら100行へ実行を移します．Iが9以下なら次の行を実行します．
		IF I>9 THEN #2	Iが9より大きいならプログラムエリアP 2へ実行を移します．Iが9以下なら次の行を実行します．

用　途	コマンド	使 用 例	使 用 例 意 味
条件処理	IF 条件式 ;	IF $="Y"; K=K+S	条件式が成立すれば；の後に書かれた命令，式を実行する． $がYならばK=K+Sの代入式を実行する．$がY以外なら次の行を実行する．
		IF A=0 ; GOSUB 200	Aが0ならばサブルーチンを200行から実行する． サブルーチンはTHENでは実行できません．
データをキーから入力	INPUT	INPUT "KAZU", K	データ（数値，文字）をキーから入力する． KAZU?と表示する．
		INPUT L, M	？を表示するのでデータを入力．データは変数Lに入力される．続いて変数Mへ入力するために？を表示する． データはEXEを押すまで入力可
	KEY	A$=KEY	KEYを実行したとき押されているキーがあればA$へ代入する． 入力できるのは1文字．
角度単位指定			
デグリー	MODE 4	MODE 4	角度単位を度に指定．DEGを表示．
ラジアン	MODE 5	MODE 5	角度単位をラジアンに指定．RADを表示．1 rad=180°/π
グラジアン	MODE 6	MODE 6	角度単位をグラジアンに指定．GRAを表示．1 gra=90°/100
プリンタの使用指定	MODE 7	MODE 7	MODE 7を実行した後のPRINT文はプリンタへ出力される．
使用解除	MODE 8	MODE 8	MODE 8を実行するとPRINT文のプリンタへの出力を解除し，表示部へ表示する．

用　途	コマンド	使 用 例	使 用 例 意 味
ループ命令の折り返し点	NEXT	NEXT I	NEXT 変数名． 変数名はFORの次に書いたものと同じ． FORでループが始まり，NEXTを実行すると変数に増分を足し，終値を越えていなければFORの次の命令，式を実行する． 終値を越えていたらNEXTの次の命令，式を実行する．
データを出力	PRINT		MODE 7が指定されていたらプリンタへ出力．それ以外は表示部へ出力．
		PRINT A$ PRINT A ; B PRINT A, B PRINT "SUU=" ; L	数値，文字の定数，変数を出力．；で区切れば前のデータに続いて出力．，で区切れば行を変えて出力． PRINT文の最後が；以外では表示後STOP状態なのでEXEを押してプログラムを再開．
	PRINT CSR	PRINT CSR 3 ; A$	表示部に0から11のアドレスが付いているのでCSRで指定してデータの先頭を表示する位置を決める．
サブルーチンの終わり	RETURN	RETURN	GOSUBで指定した行から実行を始めたサブルーチンの終わり．
データの有効桁数指定	SET E	SET E3	SET E3を指定した後の数値データは有効桁数3桁の指数出力．
小数点以下の桁数指定	SET F	SET F4	SET F4を指定した後の数値データは小数点第4位まで出力．
桁数指定の解除	SET N	SET N	SET EとSET Fの指定を解除

用途	コマンド	使用例	使用例意味
プログラム実行の中断	STOP	STOP	プログラムの実行を一時中断する。変数の表示，計算などを行なった後にEXEキーを押すことでプログラムを再開する。
変数のクリア	VAC	VAC	数値変数を0，文字変数をヌルに設定。 拡張されている変数エリアも含めすべての変数をクリア。
オプション用	SAVE	SAVE 〔"ファイル名"〕	現在指定されているプログラムエリアのプログラムをテープに記録する。
	LOAD	LOAD 〔"ファイル名"〕	現在指定されているプログラムエリアにテープからプログラムを呼び戻す。
	SAVE A	SAVE A 〔"ファイル名"〕	全プログラムエリアのプログラムを同時にテープに記録する。
	LOAD A	LOAD A 〔"ファイル名"〕	全プログラムエリアにテープから同時にプログラムを呼び戻す。
	PUT	PUT 〔"ファイル名"〕 変数	変数内のデータをテープに記録する。
	GET	GET 〔"ファイル名"〕 変数	変数にテープからデータを呼び戻す。
	VER	VER 〔"ファイル名"〕	テープに記録されているプログラムまたはデータが正しく記録されているかチェックする。

〔 〕内は省略可

エラーメッセージ一覧表

エラーコード	意味	エラー原因	対策
1	メモリーオーバーまたはシステムスタックオーバー	●ステップ数不足でプログラムが書き込めない。またはメモリーが増設できない。 ●計算式が複雑すぎてスタックオーバーしている。	●不要なプログラムをクリアするか，メモリー数を減らす。 ●数式を分割して簡単にする。
2	構文エラー	●プログラム等に書式上の誤りがある。 ●代入文等で左辺の型式と右辺の型式が異なる。	●入力したプログラム等の誤りを修正する。
3	数学的エラー	●数式の演算結果が10^{100}以上の場合。 ●数値関数の入力範囲外の場合。 ●結果が不定または不能となる場合。	●演算式またはデータを修正する ●データを判断する。
4	未定義行番号エラー	●GOTO文，GOSUB文の指定行番号がない。	●指定行番号を修正する。
5	引数エラー	●引数を必要とするコマンド，関数において，引数が入力範囲外の場合。	●引数の誤りを修正する。
6	変数エラー	●増設されていないメモリーを使おうとした。 ●同一メモリーを数値変数と文字変数に同時に使おうとした。	●適切にメモリーを増設する。 ●同時に同一メモリーを文字変数，数値変数として使わないようにする。
7	ネスティングエラー	●サブルーチン実行中以外でRETURN文が出てきた場合。 ●FORループ中以外でNEXT文が出てきたり，FOR文に対するNEXT文の変数が異なる場合。 ●サブルーチンのネスティングが8段をこえた場合。 ●FORループのネスティングが4段をこえた場合。	●不必要なRETURN文やNEXT文を取る。 ●サブルーチンやFOR･NEXTループをレベル内にする。
9	オプションエラー	●テープレコーダが接続されていないのに，SAVEまたはPUTコマンドを実行した場合。 ●LOADまたはGETコマンドにより入力された信号がわれており，読み込めない場合。 ●プリンタの充電が十分でない場合。 ●プリンタの紙づまり。	●テープレコーダを接続する。 ●テープレコーダの再生ボリュームを下げる。 ●テープレコーダのTONEを中位に調整する。 ●テープをかえる。 ●テープレコーダのヘッドをクリーニングする。 ●プリンタを充電する。 ●プリンタの紙づまりを直す。

さくいん

■アルファベット順

■五十音順

キャラクター表

	スペース	＋	−	＊	／	↑	!	”	＃	$	＞	≧	＝	≦	＜	≠
数字	0	1	2	3	4	5	6	7	8	9	.	π	)	(	E⁻	E
英大文字	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	Q	R	S	T	U	V	W	X	Y	Z						
英小文字	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
	q	r	s	t	u	v	w	x	y	z						
記号	?	,	;	:												
グラフィック	○	Σ	。	△	@	×	÷	♠	←	♥	♦	♣	μ	Ω	↓	→
記号	%	¥	□	[	&	_	’	·	]	■						

表は左から右へ，次の段の左から右へと小さい順に並んでいます．

一番小さいキャラクターは，スペースで，一番大きいキャラクターは，グラフィク記号の■（拡張モードにし，ⓈⓏを押すと表示される）です．

●監修

渡辺　茂　東京大学名誉教授．都立工科短期大学学長．日本マイコンクラブ会長．著作多数．

●執筆

柿園昭俊　1955年鹿児島県生れ，産業能率短大情報処理科，音響メーカーを経て，現在ＳＣＣ取締役ＳＥ，原価管理・ＩＥ等のシステム開発に従事．

町田正照　タツノ化学，産能短大を経て現在ＳＣＣ取締役ＳＥ．オフコン等による生産管理に従事．

將司欣枝　ＳＣＣ取締役DESIGNER．

ＳＣＣ　㈱シンユーコンサルティング．東京都墨田区吾妻橋1－18－2　代表取締役　新妻雄次　大型機からパソコンまでの専門技術者を擁するシステムハウス．（TEL. 03－621－0891）

昭和58年3月1日初版第1刷発行
昭和58年7月15日初版第3刷発行

監修
渡辺　茂
著者
柿園昭俊
検印廃止

発行者
片岡　巌
発行所
技術評論社
東京都千代田区平河町1-4-12
相互第6ビル　〒102
TEL　03－262－9351（代）
振替口座　東京0－76098

印刷　萬友社
製本　村上製本

カシオ計算機株式会社営業本部

東京都新宿区西新宿2—6　新宿住友ビル
(〒160)　☎03-347-4811(代表)

カシオ計算機サービスセンター

旭川	0166-23-8580	〒070	旭川市七条通り8丁目
札幌	011-231-2343	〒060	札幌市中央区南一条西12丁目
釧路	0154-24-8575	〒085	釧路市光陽町6—7
青森	0177-22-7466	〒030	青森市勝田2—1—12
秋田	0188-63-7690	〒010	秋田市山王2—1—40
盛岡	0196-24-2502	〒020	盛岡市本町通り3—19—6
仙台	0222-27-1404	〒980	仙台市一番町2—3—32
山形	0236-42-8018	〒990	山形市あこや町3—12—9
郡山	0249-33-5172	〒963	福島県郡山市香久池2—16—6
宇都宮	0286-34-0395	〒320	宇都宮市西大寛2—1—3
前橋	0272-53-3000	〒371	前橋市元総社町92—5
水戸	0292-25-6985	〒310	水戸市中央1—2—20
埼玉	0486-66-8567	〒330	大宮市大成町1—181
千葉	0472-43-1751	〒280	千葉市登戸町2—276
東京	03—862-4141	〒101	千代田区神田佐久間町2—23
東京	03—583-4111	〒106	港区六本木2—3—6
城南	03—787-3721	〒145	大田区上池台1—1—6
城西	03—376-3221	〒160	新宿区西新宿4—2—18
多摩	0425-23-3531	〒190	立川市錦町3—2—25
横浜	045-211-0821	〒231	横浜市中区弁天通り6—85
平塚	0463-23-2611	〒254	平塚市新宿1196
新潟	0252-41-4105	〒950	新潟市米山3—1—5
長野	0262-28-9360	〒380	長野市岡田町30—20
甲府	0552-37-6371	〒400	甲府市城東2—22—11
沼津	0559-22-8928	〒410	沼津市高島町8—11
静岡	0542-81-8085	〒420	静岡市西中原1—4—35
浜松	0534-64-1658	〒435	浜松市西塚町324
豊橋	0532-53-2515	〒440	豊橋市魚町5
名古屋	052-263-0454	〒460	名古屋市中区栄4—6—15
岐阜	0582-62-0145	〒500	岐阜市鷹見町8
三重	0592-27-5066	〒514	津市鳥居町191
富山	0764-22-2251	〒930	富山市白銀町2—1
金沢	0762-37-8511	〒920	金沢市諸江町下丁93—1
京都	075-351-1161	〒600	京都市下京区五条通り堀川東入ル
大阪	06—362-8181	〒530	大阪市北区南森町2—1—20
奈良	0742-24-3811	〒630	奈良市西木辻町200—62
和歌山	0734-31-7807	〒640	和歌山市九家の丁5
神戸	078-392-4123	〒650	神戸市中央区北長狭通り4—4—18
岡山	0862-41-8471	〒700	岡山市西古松406—2
松江	0852-25-1311	〒690	松江市嫁島町173
福山	0849-24-2830	〒720	福山市南本庄町2—130
広島	082-263-1090	〒730	広島市稲荷町4—1
山口	0835-22-6164	〒747	防府市戎町1—10—16
高松	0878-62-5240	〒760	高松市亀岡町9—16
松山	0899-45-2234	〒790	松山市平和通り1—1—5
福岡	092-411-2684	〒812	福岡市博多区博多駅南1—2—15
長崎	0958-61-8084	〒852	長崎市宝栄町2—26
熊本	0963-67-0650	〒862	熊本市健軍4—1—5
鹿児島	0992-56-3575	〒890	鹿児島市上荒田町30—18

規格

型式	PB－300
基本計算機能	負数，指数，カッコを含む四則計算（加減・乗除の優先順位判別機能つき）
組込関数機能	三角・逆三角関数（角度単位は度・ラジアン・グラジアン），対数・指数関数，開平，べき乗，整数化，整数部除去，絶対値，符号化，有効桁数指定，小数以下指定，乱数，π

関数桁容量	入力範囲	答の精度
$\sin x$，$\cos x$，$\tan x$	$\lvert x\rvert < 1440°$（8πrad, 1600gra）	10桁目±1
$\sin^{-1} x$，$\cos^{-1} x$	$\lvert x\rvert \leqq 1$	〃
$\tan^{-1} x$		〃
$\log x$，$\ln x$	$x>0$	〃
e^x	$x=1$	〃
$\sqrt{x}$	$x \geqq 0$	〃
x^y（$x\uparrow y$）	$x>0$	〃

コマンド	INPUT, PRINT, GOTO, FOR·NEXT, IF-THEN, GOSUB, RETURN, STOP, END, RUN, LIST, LIST A, MODE, SET, VAC, CLEAR, CLEAR A, DEFM, SAVE, SAVE A, LOAD, LOAD A, PUT, GET, VER
プログラム関数	KEY, CSR, LEN, MID, VAL
計算範囲	$\pm 1\times 10^{-99} \sim \pm 9.999999999\times 10^{99}$および ，　内部演算は仮数部12桁を使用
プログラム方式	ストアード方式
プログラム言語	BASIC（ベーシック）
ステップ数	最大 1,568ステップ
組込プログラム数	最大10組（P0～P9）
メモリー数	26～最大222メモリーおよび専用文字変数（$）
スタック数	サブルーチン　8段 FOR·NEXTループ　4段 数値　6段 演算子　12段

表示方式および内容	仮数部10桁（負符号含む），または仮数部8桁（負数7桁）＋指数部2桁，EXT，Ⓢ，RUN，WRT，DEG，RAD，GRA，TR，PRT，STOPの各状態表示付
表示素子	12桁ドットマトリクス液晶
主要素子	C－MOS VLSI 他
電源	リチウム電池（CR-2032）2個使用
消費電力	最大 0.02W
電池寿命（連続使用）	本体のみ　約100時間（計算のみ） 約 40時間（印字中） FA-3使用時　約 90時間（計算を含む） 約 40時間（印字を含む）
オートパワーオフ	約7分
●プリンタ部	
印字方式	感熱記録方式（ノンインパクト型）
印字桁数	20桁
印字速度	約1.0行/秒
文字構成	5×7ドット
ロールペーパー	紙幅38㎜ 外径16㎜以内の感熱記録紙
電源	充電式電池（内蔵）
電池寿命	完全充電状態で約3,000行
消費電力	4.1W
■共通部	
使用温度	0℃～40℃
大きさ・重さ	幅173 奥行90 高さ20㎜，重さ258g（電池込み）
付属品	ソフトケース，ロールペーパー，充電器（CHA-1）
別売品	ロールペーパー〈TRP－16〉￥300（5巻）